

ARI Research Note 99-25

Modification of the Computer Adaptive Screening Test (CAST) for Use by Recruiters in All Military Services

James R. McBride and R. Ross Cooper
Human Resources Research Organization

Selection and Assignment Research Unit
Michael G. Rumsey, Chief

April 1999



U.S. Army Research Institute
for the Behavioral and Social Sciences

Approved for public release; distribution is unlimited.

DTIC QUALITY INSPECTED 2

19990427 026

**U.S. Army Research Institute
for the Behavioral and Social Sciences**

A Directorate of the U.S. Total Army Personnel Command

**EDGAR M. JOHNSON
Director**

Research accomplished under contract
for the Department of the Army

Human Resources Research Organization

Technical review by

Michael G. Rumsey

NOTICES

DISTRIBUTION: This Research Note has been cleared for release to the Defense Technical Information Center (DTIC) to comply with regulatory requirements. It has been given no primary distribution other than to DTIC and will be available only through DTIC or the National Technical Information Service (NTIS).

FINAL DISPOSITION: This Research Note may be destroyed when it is no longer needed. Please do not return it to the U.S. Army Research Institute for the Behavioral and Social Sciences.

NOTE: The views, opinions, and findings in this Research Note are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision unless so designated by other authorized documents.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY (Leave Blank)

2. REPORT DATE
April 1999

3. REPORT TYPE AND DATES COVERED
Final Report, September 1996 - June 1997

4. TITLE AND SUBTITLE

Modification of the Computerized Adaptive Screening Test (CAST)
for Use by Recruiters in All Military Services

5. FUNDING NUMBERS

MDA903-93-D-0032
(Delivery Order 0054)

2Q162785A791

2210C1

6. AUTHOR(S)

James R. McBride, and R. Ross Cooper

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Human Resources Research Organization (HumRRO)
66 Canal Center Plaza, Suite 400
Alexandria, Virginia 22314

8. PERFORMING ORGANIZATION REPORT NUMBER

FR-WATSD-97-24

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Army Research Institute for the
Behavioral and Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

Research Note 99-25

11. SUPPLEMENTARY NOTES

Contracting Officer's Technical Representative, Ms. Frances Grafton

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release, distribution
unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The Computerized Adaptive Screening Test (CAST) was designed to predict performance on the Armed Forces Qualification Test (AFQT). It includes two subtests: Word Knowledge (WK) and Arithmetic Reasoning (AR). CAST has been used by Army recruiters since the early 1980's to prescreen enlistment prospects. The Joint Recruiting Information Support Systems Program Management Office (JRISS PMO) program requested modifications to CAST to make it suitable for use by recruiters in all of the U.S. military services. This report documents the development of CAST, Version 5.

14. SUBJECT TERMS

CAST, Adaptive testing, AFQT

15. NUMBER OF PAGES

143

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT

Unclassified

18. SECURITY CLASSIFICATION OF
THIS PAGE

Unclassified

19. SECURITY CLASSIFICATION OF
ABSTRACT

Unclassified

20. LIMITATION OF ABSTRACT

UL

**CONTRACT FOR MANPOWER AND PERSONNEL RESEARCH
AND STUDIES (COMPRS)
FOR THE U.S. ARMY RESEARCH INSTITUTE (ARI)**

**MODIFICATION OF THE
COMPUTERIZED ADAPTIVE SCREENING TEST (CAST)
FOR USE BY RECRUITERS IN ALL MILITARY SERVICES**

FINAL TECHNICAL REPORT

**James R. McBride
R. Ross Cooper**

Submitted to:

**U.S. Army Research Institute for the Behavioral and Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333**

**Contract Number MDA903-93-D-0032
Delivery Order 0054
Clin No. 0002AB**

September 10, 1997

HUMAN RESOURCES RESEARCH ORGANIZATION

66 Canal Center Plaza, Suite 400 • Alexandria, Virginia 22314 • (703) 549-3611

Modification of the Computerized Adaptive Screening Test (CAST) for Use by Recruiters in all Military Services

Table of Contents

	<u>Page</u>
Introduction	1
CAST 5 Psychometrics	2
Background	2
Changes Introduced with CAST 5	3
CAST's Prediction of ASVAB Performance	4
CAST 5 Software	6
System Requirements	6
Test Administration Program	6
Provisions for User Control of Some CAST Features	6
Item Bank Utility Program	10
CAST 5 Software Files	10
Database Descriptions	13
The CAST Database	13
The EXAMINEE Database	16
The SECURITY Database	17
Bitmap Image Files	18
References	19
 <u>Appendixes</u>	
A: CAST 5 for Windows Users' Guide	A-1
B: CAST 5 Source Code	B-1

List of Tables

Table 1. The CAST.INI file, with values included in the first release of Cast 5	9
Table 2. Critical Files Installed or Updated by the CAST 5 SETUP Program	11
Table 3. A Directory Listing of the Source Code Files Used to Compile CAST 5 Version 1.15	12

Modification of the Computerized Adaptive Screening Test (CAST) for Use by Recruiters in all Military Services

Introduction

This report describes Version 5 of CAST, the Computerized Adaptive Screening Test. CAST is a screening test for use by military recruiters to forecast a prospect's performance on the Armed Forces Qualification Test (AFQT), as well as on two of its components, the Word Knowledge (WK) and Arithmetic Reasoning (AR) sections of the Armed Services Vocational Aptitude Battery (ASVAB).

CAST was originally developed for use by Army recruiters, as described by Sands and Gade (1983). Since its introduction in the early 1980's, it has undergone one major psychometric revision, and has been modified for use on a succession of microcomputer models. Wise, McHenry, Chia, Szenas and McBride (1989) described the psychometric revision, which was first implemented in Version 2 of CAST. Prior to the development summarized in this report, the most recent version of CAST was Version 4, which was described by Park and Dunn (1991). CAST 4, like earlier versions, was used only by Army recruiters.

The development of CAST 5 was sponsored by JRISS, the Joint Recruiting Information Support System, for use by recruiters of all of the Armed Services. CAST 5 updates the previous version in several significant ways. For one, CAST 5 has been developed specifically for use on computer systems with Intel Pentium and compatible processors, under the Microsoft Windows 95 or Windows NT operating environment; it is not compatible with earlier computer models or previous operating systems. Aside from this, the most noticeable difference between CAST 5 and previous versions is the user interface. Previous versions of CAST presented instructions, test questions, and score reports in a monochrome display format, using the computer keyboard as the exclusive means of entering data and answering test questions. In contrast, CAST 5 uses a color graphic user interface, with the option of using a computer's keyboard or its "mouse" pointing device to navigate through the program.

CAST 5 is also designed to predict AFQT scores more accurately than previous versions. To make this possible, CAST 5 administers more test questions than CAST 4, and adjusts test length to enhance accuracy at the 31st and 50th AFQT percentiles. Because it administers more test questions, CAST 5 can be expected to take somewhat longer to administer than previous versions. Even with the additional length, CAST 5 will generally take less than 25 minutes to administer.

The remainder of this report describes CAST 5 in more detail. The report is aimed at two different audiences: (1) personnel researchers and psychometricians, who will be interested in CAST's measurement and score prediction features; and (2) information systems managers and technicians, who will be more interested in features of the CAST 5 software system. Although the report is not intended to be a guide for CAST 5 test administration, in the interests of completeness, a Users' Guide is included as Appendix A.

This report contains three major sections. First is a Psychometrics section; beginning with some background on previous CAST versions, it treats CAST 5 as a psychometric application, and proceeds to describe psychometric features and issues that are specific to CAST 5. Second is a System Description section; it treats CAST 5 as an automated data system, and includes technical information that will be useful in the operation, maintenance and revision of its software. Third are the Appendixes. These include the Users' Guide, and a copy of the CAST 5 source code.

CAST 5 Psychometrics

This section addresses certain aspects of CAST 5 as a system for mental measurement and for forecasting examinees' scores on the AFQT and on ASVAB's WK and AR tests.

Background

CAST is a battery of two computer-administered adaptive tests -- WK and AR. The tests are called "adaptive" because each of them uses the computer software to tailor test difficulty to the examinee's ability by selecting test questions one at a time, contingent on the test-taker's performance. Scores on these two CAST subtests are used as the basis for forecasting the examinee's AFQT score, using relationships between WK, AR, and AFQT scores that were established in previous CAST research.

CAST is not the only screening test available to recruiters for the purpose of predicting prospects' AFQT performance. The Enlistment Screening Test (EST) was designed for the same purpose (Mathews & Ree, 1982). Unlike CAST, the EST was designed as a paper-and-pencil test. Although research has shown that CAST is somewhat more accurate than EST for predicting AFQT scores, the chief advantage of CAST over EST is efficiency. CAST is adaptive; adaptive tests typically achieve the measurement precision of conventional tests in half the latter's length. Indeed, previous versions of CAST were as short as 15 questions in length and took less than 20 minutes to administer. This is in contrast with EST's much greater test length, and its administration time of 45 minutes. Recently, a computer-administered version of the EST has appeared, called the Computerized Enlistment Screening Test (CEST). Like CAST, it is computer-administered. CEST is not adaptive, however, and is considerably longer than CAST, both in terms of the number of test items given and test duration.

CAST was originally developed for use in the Army's JOIN (Joint Optical Information Network) computer system. CAST Version 1 administered 10 WK and 5 AR test questions. These two tests were selected adaptively from data banks containing fewer than 100 questions in each subject area. The first revision of CAST, reported by Wise et al. (1989), developed much larger banks of test questions to replace the earlier ones. All subsequent versions of CAST have used these two expanded item banks, which contain 257 WK and 254 AR test questions. CAST Versions 2 through 4 administered a few additional test questions to enhance measurement precision for individuals performing below specified score thresholds.

CAST 5 preserves most of the psychometric features of its predecessor, CAST 4. For example, it uses the same (1) test item banks (257 WK items and 254 AR items calibrated by Wise et al., 1989); (2) sequential technique for updating ability estimates after each test item (the Bayesian sequential updating procedure described by Owen, 1975); (3) parameters of the normal prior distributions on ability (mean 0, variance 1); and (4) adaptive item selection criterion (local values of test item information functions).

Changes Introduced with CAST 5

CAST 5 differs from CAST 4 psychometric features in the following ways: (1) longer test length for both the WK and AR subtests; (2) use of adaptive test length during the AR test; (3) imposition of a time limit on the AR test; (4) increased use of randomization in the item selection/exposure control process. Each of these differences is described in more detail below.

Longer test length. CAST 4 had developed a reputation for inaccuracy among some recruiters. Although there are no systematic data to support that reputation, and ample data (Knapp & Pliske, 1987; Wise et al., 1989) to refute it, it had become part of the lore of CAST. In response to that lore, other means of screening prospects were said to be preferred over CAST. These include the paper-and-pencil EST and the computer-administered CEST.

To counter the impression that CAST is less accurate than the alternatives, priority in the development of CAST 5 was given to improving CAST's precision. One means of improving measurement precision is to increase test length; this has been done in CAST 5. The length of the WK test in CAST 5 has been set to 15 items; in contrast, CAST 4 gave 10 to 15 WK items, depending on the examinee's performance. CAST 5 administers 7 to 12 AR items; CAST 4's AR tests were 5 to 10 items in length.

Variable test length. The length of the AR test is adaptive. The minimum length is set by the ARLength argument in CAST.INI; this value was 7 at release time. The maximum length is set by the MaxARLength argument. After 7 AR items have been administered, the test will stop unless the predicted AFQT score is in the critical range specified by two other arguments in CAST.INI: LowAFQTCutoff and HighAFQTCutoff. Following the 7th and subsequent items, the AR test will continue, until MaxARLength items have been given, if the predicted AFQT score is between the upper and lower bounds of the critical range described below. At release time, MaxARLength was set at 12 items. At any point after the 7th and subsequent AR items, if the updated value of the predicted AFQT score falls outside the critical interval, the test will stop. Thus, the length of the AR test can be 7, 8, 9, 10, 11, or 12 items.

The new CAST test lengths reflect previous research (Knapp, 1987) that established CAST's predictive validity for all combinations of WK test lengths from 5 to 15 items, and AR test lengths from 5 to 10 items. That research demonstrated that the multiple correlation of CAST with AFQT increased from .82 to .84 as WK and AR test lengths were increased from 5 WK and 5 AR items to 15 WK and 10 AR items. As test length increased, Knapp reported a corresponding decrease from 14 to 13 in the standard error of CAST estimates of AFQT scores. By making the length of the AR test contingent on the predicted value of the AFQT score, CAST

5 aims to improve the precision of AFQT score estimates in the neighborhood of the lower and upper critical points specified in CAST.INI. At release time, the two points of interest were the 31st and 50th AFQT score percentiles. Allowing for a 13-point standard error of estimate, the critical range becomes 18 to 63. During CAST 5's AR test, the predicted AFQT score is recalculated after the 7th and subsequent test items; if it is between 18 and 63, additional AR test items will be administered until the maximum length is reached.

Time limit. CAST 4 was administered without time limits; in some cases, recruiters observed that some prospects had very long test times, largely due to slow work on the AR portion of the test. Sponsors of CAST 5 development requested a time limit of 25 minutes on that part of the test. Accordingly, the CAST 5 software includes a timer; the AR test is stopped after 25 minutes have elapsed on that part of the test.

When the AR test time limit is reached, a penalty function is applied to the AR test score to prevent high AR scores that could result as an artifact of the adaptive ability estimation procedure. Segall, Moreno, Bloxom and Hetter (in press) reported development of a similarly motivated procedure for use in the computerized adaptive version of the ASVAB (CAT-ASVAB). In CAST 5, if the examinee has not completed at least 7 items after 25 minutes have elapsed, the ability estimate is updated as if the examinee gave wrong answers to the unreached portion of the 7-item minimum test length; this is a somewhat more stringent penalty function than the one used in CAT-ASVAB, where the penalty computation assumes random responses, rather than wrong ones.

Item selection/item exposure control. In CAST 5, the criterion for adaptive selection of the next test item is identical to CAST 4, with the following exception: In CAST 5, the next item selected is always randomly chosen from a set containing the 5 currently optimal items. In CAST 4, the size of the candidate item set diminished from 5 to 1 item as the test progressed; after the fifth item in each test, the optimal item was always selected. This resulted in predictable sequences of test items in certain circumstances, particularly after answering the first 4 items all wrong or all right. Selecting an item at random from among the best set of 5 will reduce the incidence of repeated sequences of the same questions.

CAST 5 also differs from CAST 4 in terms of the mechanism used for item exposure control. At each stage in CAST 4, each item in the list of locally optimal test items was excluded from use later in the same CAST test. CAST 5 excludes only those items that are actually administered -- not the other 4 candidate items at each stage. This will effect a small improvement in psychometric precision within CAST's WK and AR subtests, at the expense of item exposure control. Thus CAST 5's only mechanism for item exposure control is random selection from the locally best 5-item set at each stage.

CAST's Prediction of ASVAB Performance

AFQT percentile prediction. CAST 5 uses the same formula as CAST 4 for forecasting the examinee's AFQT score: a linear multiple regression equation for estimating the AFQT

percentile from the examinee's scores on the two adaptive CAST tests, WK and AR. The equation is as follows:

$$\text{CAST 4 Forecast AFQT Percentile} = 14.41 \text{ WK} + 11.18 \text{ AR} + 42.775$$

In this equation, WK and AR are CAST's internal estimates of examinee ability on the respective tests. The scale used for WK and AR is the "theta" metric. CAST scores on that metric typically vary between -3 and +3, with mean 0 and approximately unit variance.

Since the introduction of the original version of CAST, linear multiple regression has been used as the basis for AFQT prediction. Early CAST technical reports (e.g., Knapp, 1987; Knapp & Pliske, 1986) documented the development and validation of CAST regression parameters, but did not list their values. The most recent report of research to develop an equation for predicting AFQT scores from CAST was that of Wise et al. (1989), who developed CAST Version 2, which included the current CAST WK and AR item banks. With the introduction of the new item banks it was necessary to develop a new regression equation; Wise et al. reported the details of that development, and listed the parameters of their regression equation as:

$$\text{Wise et al. Forecast AFQT Percentile} = 9.50 \text{ WK} + 10.27 \text{ AR} + 56.11$$

This equation states the regression parameters developed for CAST 2. It predicts substantially different AFQT score values than does the CAST 4 equation above. For example, if the values of both WK and AR are 0, the Wise et al. equation predicts an AFQT score of 56, while the CAST 4 equation predicts a 43 (both predictions have been rounded to the nearest integer). Even larger differences occur at lower WK and AR values. The origin of the CAST 4 regression parameters is not documented, although their values are listed in the report by Park and Dunn (1991).

AFQT category prediction. In addition to predicting point values of examinees' AFQT percentile scores, CAST estimates the probability that the examinee's AFQT score will fall into each of three AFQT score bands¹. These are as follows:

<u>Category</u>	<u>AFQT Score Band</u>
I - IIIa	50 - 99
IIIb - IVa	31 - 49
IVb - V	1 - 30

¹ Although CAST 5, like CAST 4 before it, calculates these AFQT category probabilities, the probability values are not included in CAST 5 score reports. They are, however, recorded in examinee data files.

Thus, the first of these probability statements addresses the likelihood that the examinee will attain a score of 50 or above. This is a critical value in recruiting, because enlistment incentives are sometimes contingent on AFQT scores in the top half of the percentile scale, and because the proportion of enlistees in that range is used as one indicator of the overall quality of military personnel accessions.

The second two AFQT score ranges above are also important in recruiting because accession policy often prohibits enlistment of candidates with AFQT scores below the 31st percentile. CAST's probability statements allow a recruiter to judge the odds that a candidate will score above or below the 31st percentile threshold.

These probability statements are computed by means of logistic regression equations, which express the probability of these discrete events as functions of CAST WK and AR scores. The regression equations were developed by Wise et al. (1989); the values of the logistic regression parameters are listed in that report.

CAST 5 Software

System Requirements

CAST 5 is a 16-bit application program designed specifically for use on computers using Intel Pentium and equivalent microprocessors under the Microsoft Windows 95 or Windows NT operating environment. As released, CAST 5 is not compatible with earlier Windows versions, and has not been tested for compatibility with non-Pentium processors.

Test Administration Program

CAST 5's test administration program was written with the Microsoft Visual Basic system, version 4. The source code was compiled into executable form using the version 4 Visual Basic compiler, and an installation program was created using the Microsoft SETUP Wizard. Appendix B contains complete source code listings for the version delivered to JRISS (CAST 5 version 1.15).

Provisions for User Control of Some CAST Features

Some of the features of CAST can be changed by the user without the need to revise the CAST software. These features are controlled by specifications recorded in CAST.INI, an ASCII text file included with CAST 5. The user can change them simply by editing the values recorded in CAST.INI.

Table 1 displays the contents of CAST.INI at release time. Each line in the display is numbered; the function of each numbered line is discussed below.

Lines 1 and 2 contain the software version and its release date; these lines should be updated with each revision of CAST.

Lines 3 and 4 specify path and file names for respectively: (1) the database file containing CAST 5 instructional screen text, test item text, test item parameters, and adaptive item selection tables (information tables); and (2) the database file to be used for detailed test records. The item database file contains both regular (calibrated) test items and experimental test items. The test records database stores data recorded at the item response-level. Both of these files must be specially formatted for use by CAST.

Line 5 specifies the data path CAST 5 will use for access to other data files it needs. Line 6 specifies the file containing the bitmap image that will be used as the background of the main menu screen.

Lines 7 through 10 specify colors used in CAST 5 displays, including the display foreground and background colors, and the normal and highlighted colors used to display response alternatives.

Lines 11 and 12 specify the positions in the test item database file containing the test used in the initial and final instructional screens. Line 13 specifies the character font size, in points.

Line 14 sets the time limit, in minutes, used for the AR subtest.

Lines 15 through 17 are the values of logical toggles that control certain software options; if the value is 0, the toggle is false; -1 sets it to true. When these toggles are set to true, the options are enabled. Line 15 toggles a debug feature. Line 16 toggles test item review; when this toggle is set to -1, CAST displays every test question in the database, up to the limits specified in lines 18 and 19; when it is set to 0, CAST is administered normally. Line 17 toggles on or off the display of detailed psychometric data after each test item; this feature is used to check and debug adaptive item selection and ability updating calculations.

Lines 18 and 19 specify the number of adaptive test items in the database for WK and AR, respectively. Lines 37 and 38 specify the number of experimental test items for WK and AR.

Lines 20 through 26 govern the length of the two adaptive tests. Line 20 sets the minimum number of WK items to be administered; line 21 sets the minimum for AR. Lines 22 and 23 govern the administration of additional adaptive WK items; the number is specified in line 22, and will be administered if the WK ability estimate falls below the threshold specified in line 23. Lines 24 through 26 govern the administration of additional AR items; these will be administered if the predicted AFQT score lies between specified lower and upper bounds. Line 24 specifies the maximum number of additional AR items; lines 25 and 26 specify the critical lower and upper predicted-AFQT boundaries.

Lines 27 through 32 specify the parameters for two logistic regression equations. The first equation estimates the probability that the examinee's AFQT score will be in AFQT categories 1 through 3a; that is, between 50 and 99 inclusive. The second equation estimates the probability the AFQT score will be in category 3b or 4a: between 31 and 49. Input variables to these two logistic regression equations are CAST's internal estimates of ability (θ) in the WK and AR domains.

Lines 33 through 35 specify the parameters of the linear multiple regression equation CAST uses to predict the examinee's AFQT percentile score. Input variables to these two logistic regression equations are the same ones used for the logistic regression: CAST's internal estimates of ability (θ) in the WK and AR domains.

Lines 36 and subsequent lines contain parameters that control the optional administration of experimental test items in the CAST tests. Experimental items can be embedded in WK, AR, or in both tests. Lines 36 and 37 specify the size of the experimental WK and AR item banks, respectively. Lines 38 and 39 specify the number of experimental WK and AR items, respectively, to be administered. If either or both of line 38 and 39 is greater than 0, the following lines must contain specifications for the serial positions of the experimental items, and the number of experimental items at each serial position.

Table 1. The CAST.INI file, with values included in the first release of Cast 5.

<u>Line</u>	<u>Contents</u>	<u>Description</u>
1	5.15	Software version number
2	04/23/97	Version release date
3	c:\wincast\cast.mdb	Test questions database file
4	c:\wincast\examinee.mdb	Test results database file
5	c:\wincast\	Path name for other files
6	c:\wincast\jrislogo.bmp	Bitmap file for main menu screen
7	0 Forecolor	Foreground color control code
8	7 Backcolor	Background color control code
9	4 Distractor Color	Color used for response alternatives
10	11 Highlight Color	Color used to highlight alternatives
11	1 IntroScreen Start	"Item number" of first instructions text
12	5 IntroScreen End	"Item number" of last instructions text
13	12 Font Size	Point size for test item display fonts
14	25 AR Time Limit	Time limit (minutes) for AR test
15	0, Debug flag	Toggle (0/1) debug mode
16	0, Display all items?	Toggle (0/1) display of entire item bank
17	0, Display psychometrics?	Toggle (0/1) for psychometric data display
18	257, WK item bank size	Number of WK items in CAST database
19	254, AR item bank size	Number of AR items in CAST database
20	15, WK minimum length	Minimum length of WK test
21	7, AR minimum length	Minimum length of AR test
22	0, WK additional length	Number of extra WK items permissible
23	0.00, Critical value for WK	Threshold value for extra WK items
24	5, AR additional length	Number of extra AR items permissible
25	18, Critical AFQT minimum	Lower AFQT bound for extra items
26	63, Critical AFQT maximum	Upper AFQT bound for extra items
27	-2.57, LR WK parameter 1	WK logistic regression value for CAT 1-3a
28	1.96, LR AR parameter 1	AR logistic regression value for CAT 1-3a
29	0.27, LR constant 1	Logistic regression constant for CAT 1-3a
30	-2.00, LR WK parameter 2	WK logistic regression value for CAT 3b
31	1.49, LR AR parameter 2	AR logistic regression value for CAT 3b
32	2.36, LR constant 2	Logistic regression constant for CAT 3b
33	14.41, LMR WK parameter	WK linear regression parameter for AFQT AR
34	11.18, LMR AR parameter	linear regression parameter for AFQT
35	42.775, LMR constant	Linear regression constant for AFQT
36	96, WK exper. item bank size	Number of experimental WK items in bank
37	96, AR exper. item bank size	Number of experimental AR items in bank
38	0, Give N exper. WK items	Number of experimental WK items to give
39	0, Give N exper. AR items	Number of experimental AR items to give
40	0, 0 Position exper. WK items	Serial position of experimental WK items
41	0, 0 Position exper. AR items	Serial position of experimental AR items

Item Bank Utility Program

CAST 5 draws its test questions from two banks of operational questions, and two banks of experimental questions. The operational banks contain 257 WK and 254 AR questions; these questions are the source of adaptive CAST tests administered to individual examinees. CAST is also capable of embedding a small number of experimental items within each test; such items do not affect the CAST test scores, but response data are recorded for research purposes. CAST 5 includes 98 items in each of two experimental banks: WK and AR.

All of the CAST test questions -- operational as well as experimental -- are contained in an encrypted Microsoft Jet database file, CAST.MDB. The same file contains the text of CAST's instructional screens and help screens. The file contains tables of CAST item parameters: difficulty, discrimination, and lower asymptotes are recorded for each item. Additionally, CAST.MDB contains CAST's adaptive item selection tables.

A utility program (Project1.VBP) that accompanies the CAST 5 source code makes it possible to edit UCAST.MDB, the unencrypted version of CAST.MDB. This might be desirable in order to modify the text of CAST instructional or help screens, or to correct errors if any are found in the item bank text or answer keys. If it becomes necessary to make corrections to the CAST.MDB file, the corrections must be made in two stages, as follows: 1) Make corrections to UCAST.MDB, by means of the utility program. 2) Encrypt the corrected database file, saving the encrypted file as "CAST.MDB." The encryption can be done in either of two ways. One is to use the encryption/decryption utility function of Visual Basic 4. The other is to use Microsoft ACCESS Version 2 database software to encrypt UCAST.MDB, creating a revised version of CAST.MDB. In the encryption process, CAST.LDB is automatically updated.

It is important to remember that changing item text or answer keys may have a significant impact on the psychometric properties of CAST and should not be done except in the most extreme circumstances. Whenever changes are made to UCAST.MDB, creating a revised CAST.MDB, the user should edit lines 1 and 2 of the CAST.INI file to update the software version number and release date.

CAST 5 Software Files

Distributed file list. CAST 5 is distributed on a 3-disk installation package created by means of the Microsoft Setup Wizard program. The SETUP.EXE program included in that package installs CAST 5 program and support files, as well as Windows system files, on the user's computer. All CAST 5 files are to be installed in a directory named WINCAST on the host computer's C: drive. The SETUP.EXE program installs Windows system files in the Windows 95 or Windows NT directories, as needed. In general, SETUP installs Windows system files unless it finds a current version of the file already present. If one of the necessary system files is missing, SETUP installs it; if the file is present but outdated, SETUP installs the newer version.

Table 2 contains a list of the CAST 5 program, support, and system files. In general, CAST 5 will not run properly if any of its program or support files are missing from the C:\WINCAST directory, or if the system files listed are missing or outdated.

Table 2. Critical Files Installed or Updated by the CAST 5 SETUP Program.

<u>File Name</u>	<u>Path Name</u>	<u>File Date</u>	<u>File Size</u>	<u>Version</u>
[bootstrap]				
SETUP1.EXE	\$(WinPath)	1/12/1996	138144	4.0.0.2422
VSHARE.386	\$(WinSysPath)	1/12/1996	14933	3.11.0.401
STKIT416.DLL	\$(WinSysPath)	1/12/1996	5120	4.0.2422.0
VB40016.DLL	\$(WinSysPath)	1/12/1996	935632	4.0.24.22
OC25.DLL	\$(WinSysPath)	8/15/1995	536048	2.53.0.0
OLE2.DLL	\$(WinSysPath)	1/12/1996	304640	2.3.125.142
TYPELIB.DLL	\$(WinSysPath)	1/12/1996	177824	2.3.3025.1
OLE2DISP.DLL	\$(WinSysPath)	1/12/1996	164960	2.3.3023.1
OLE2PROX.DLL	\$(WinSysPath)	1/12/1996	51712	2.2.120.121
OLE2CONV.DLL	\$(WinSysPath)	1/12/1996	57328	2.1.0.1
STORAGE.DLL	\$(WinSysPath)	1/12/1996	157696	2.3.125.140
COMPOBJ.DLL	\$(WinSysPath)	1/12/1996	109056	2.3.125.142
OLE2.REG	\$(WinSysPath)	1/12/1996	28113	
OLE2NLS.DLL	\$(WinSysPath)	1/12/1996	152976	2.3.3023.1
STDOLE.TLB	\$(WinSysPath)	1/12/1996	5472	2.3.3023.1
SCP.DLL	\$(WinSysPath)	1/12/1996	12976	2.0.260.0
VAEN21.OLB	\$(WinSysPath)	1/12/1996	35200	2.0.0.5422
CTL3DV2.DLL	\$(WinSysPath)	9/9/1995	27632	2.31.0.0
[Files]				
DAO2516.DLL	\$(WinSysPath)	1/12/1996	543584	2.50.0.1626
MSAJT200.DLL	\$(WinSysPath)	8/15/1995	995136	2.50.0.1606
MSJETERR.DLL	\$(WinSysPath)	11/18/1994	11232	2.50.0.1111
MSJETINT.DLL	\$(WinSysPath)	11/18/1994	15936	2.50.0.1111
VBAJET.DLL	\$(WinSysPath)	1/12/1996	2920	2.0.0.5422
VBDB16.DLL	\$(WinSysPath)	1/12/1996	86848	4.0.24.22
WINCAST.EXE	\$(AppPath)	4/23/1997	145424	1.0.0.0
CAST.LDB	\$(AppPath)	4/23/1997	64	
CAST.MDB	\$(AppPath)	4/23/1997	262144	
CASTLOGN.BMP	\$(AppPath)	3/4/1997	185398	
ENDTEST.BMP	\$(AppPath)	4/3/1997	460918	
EXAMINEE.LDB	\$(AppPath)	4/23/1997	64	
EXAMINEE.MDB	\$(AppPath)	4/23/1997	229376	
FEEDBACK.BMP	\$(AppPath)	4/9/1997	460918	
MOVEDATA.LDB	\$(AppPath)	4/23/1997	64	
MOVEDATA.MDB	\$(AppPath)	4/23/1997	229376	
PASSWORD.BMP	\$(AppPath)	12/23/1996	185398	
SECURITY.LDB	\$(AppPath)	4/23/1997	64	
SECURITY.MDB	\$(AppPath)	4/23/1997	65536	
SYSTEM.LDB	\$(AppPath)	12/23/1996	64	
SYSTEM.MDA	\$(AppPath)	12/23/1996	98304	
CAST.INI	\$(AppPath)	4/23/1997	962	

Legend:

\$(WinPath) Path name to Windows directory
 \$(WinSysPath) Path name to Windows system files directory
 \$(AppPath) Path name to CAST 5 application program directory

Program and Source code files. The program file WINCAST.EXE contains the executable CAST 5 program. It was compiled by the Microsoft Visual Basic Version 4 system as a 16-bit application program. The source code for this program is contained in the files listed in Table 3; these files are not distributed with the SETUP program.

Table 3. A Directory Listing of the Source Code Files Used to Compile CAST 5 Version 1.15.

File Name	Size	Date	Time	Extended File Name
ABOUT FRM	3,558	03-04-97	7:59p	ABOUT.FRM
CAST BAS	43,560	10-23-96	10:07p	CAST.BAS
CONSTANT TXT	38,244	04-28-93	12:00a	CONSTANT.TXT
CONTROL BAS	12,362	04-21-97	2:22p	CONTROL.BAS
DIALOG FRM	7,817	04-18-97	8:46a	DIALOG.FRM
EXAMUTIL FRM	14,835	04-23-97	3:40p	EXAMUTIL.FRM
FEEDBACK FRM	10,265	04-23-97	3:40p	FEEDBACK.FRM
FILESELE FRM	4,618	04-17-97	10:53p	FILESELE.FRM
FORM1 FRM	486	10-30-96	12:18a	FORM1.FRM
HELP FRM	1,787	04-23-97	10:17a	HELP.FRM
HELP FRX	8	04-23-97	10:17a	HELP.FRX
ITEMWIND FRM	51,313	04-23-97	11:13a	ITEMWIND.FRM
ITEMWIND FRX	1,233,160	02-27-97	10:57a	ITEMWIND.FRX
PWORD FRM	13,133	04-01-97	11:14p	PWORD.FRM
SAMPLES FRM	5,256	04-01-97	11:59p	SAMPLES.FRM
SECURITY FRM	34,618	04-23-97	3:40p	SECURITY.FRM
SECURITY FRX	92	04-23-97	3:40p	SECURITY.FRX
VIEWEXAM FRM	1,473	04-23-97	3:40p	VIEWEXAM.FRM
VIEWEXAM FRX	6	04-23-97	3:40p	VIEWEXAM.FRX
WINCAST MAK	666	04-23-97	3:40p	WINCAST.MAK
WINCAST VBZ	3,769	04-24-97	9:55a	WINCAST.VBZ

Database files. There are three databases used in the CAST 5 (WinCast) system. These databases are named:

CAST
EXAMINEE
SECURITY

Each of these databases contains two component files. One file is designated with the file extension "MDB." The second part is designated with the file extension "LDB."

For example, the following file names will be found in the CAST system directory:

CAST.MDB
CAST.LDB
EXAMINEE.MDB
EXAMINEE.LDB
SECURITY.MDB
SECURITY.LDB

In addition to the database filenames listed above, two additional databases will also be found in the WinCast subdirectory. These are:

MOVEDATA SYSTEM

MOVEDATA files (MOVEDATA.LDB and MOVEDATA.MDB) contain a duplicate of the blank test results database. These files are provided as a means to copy or transfer test results. Instructions for these functions are included in the CAST 5 Users' Guide at Appendix A.

CAST 5 also includes two Microsoft Jet Database system files, SYSTEM.LDB and SYSTEM.MDA; these files are used for Microsoft system functions, and were not created as part of the WinCast development system.

Each of the database files listed above is encrypted using Microsoft's encryption feature provided with the 'Data Manager' utility in Visual Basic 4.0. (The encrypted files are readable only by the Microsoft ACCESS database software; that software's file encryption/decryption feature can be used to decrypt and edit CAST 5 database files.) The CAST, EXAMINEE and SECURITY databases are described below.

Database Descriptions

The CAST Database

The CAST database is designed to contain the following:

- Introduction screen text
- Help screen text
- Miscellaneous screen text
- WK sample items and feedback
- WK operational items
- WK experimental items
- WK information tables
- WK item parameters
- AR sample items and feedback
- AR operational items
- AR experimental items
- AR information tables
- AR item parameters

This file is "read-only" since the data contained within this file are used for display and calculation purposes only. This file is not modified during a test administration.

CAST fields. There are only two fields within the CAST database as follows:

(1) Field 1 - Primary: The 'Primary' field holds the record identifier. In this CAST database, all records are identified with a number between 1 and 9999. In order to locate a particular record, the 'Primary' field is used as the 'Key' for the CAST database.

(2) Field 2 - TextInfo: This field contains either text or binary information. Records containing screen text are always saved as standard 'flat' ASCII. Records containing values such as adaptive test item selection (information) tables or item parameters are stored as binary. The specific formats for information tables and parameter records are defined below.

CAST database keys. No indexes are used by the WinCast system to access the CAST database file. The 'Key' for the CAST database file is the field named 'Primary.' This 'Primary' field always contains a record number; the WinCast system uses these numbers to fetch a particular record from the CAST database file.

Information contained in the CAST database. Records with key numbers 1 through 8999 contain text data, including the text of instructions, sample (practice) questions, help screens, and banks of operational and experimental WK and AR test items. The CAST program accesses these by key numbers; contents are as follows:

Records 1 - 18	Instructions and Miscellaneous Screens
Record 101	WK Sample Item
Record 102	AR Sample Item
Records 201-207	Help Screens
Records 1001-1257	WK Items 1 through 257
Records 2001-2254	AR Items 1 through 254
Records 3001-3096	WK Experimental Items 1 through 96
Records 4001-4096	AR Experimental Items 1 through 96

Records with key numbers 9000 and greater contain binary data. These data include the item parameters and adaptive item selection tables called "information tables." The item parameters include values of parameters a, b, and c for each WK and AR test question; these are used in test scoring. The information tables contain sorted lists of item numbers; these indicate the optimal items for measurement at each of several equally spaced intervals on the WK and AR ability scales. The specific contents of the binary records are as follows:

Record 9000 - WK Item Parameters
Record 9001 - AR Items Parameters
Record 9002 - WK Information Table
Record 9003 - AR Information Table
Record 9999 - The label 'complete'

Format of the 'Parameters' Records. Each operational test item has a total of three parameters. In the WinCast system these three parameters are named 'A', 'B', and 'C'. In the CAST database file, these three parameters are laid out (in a binary format) as follows:

A parameter for item 1
B parameter for item 1
C parameter for item 1
A parameter for item 2
B parameter for item 2
C parameter for item 2
A parameter for item 3
... and so on.

Each item's three parameters require 13 bytes of storage; these 13 bytes are laid out as follows:

Bytes 1 - 4	Parameter A (the response model's slope parameter)
Bytes 5 - 9	Parameter B (the threshold, or difficulty, parameter)
Bytes 10 - 13	Parameter C (the lower asymptote, or guessing, parameter)

In order to index into this database record and fetch the parameters for a specific item number, the calculation must be performed:

Dim Ptr as Integer
$$\text{Ptr} = (13 * (\text{ItemNumber} - 1) + 1)$$

Once a pointer is correctly positioned within this record, the first 4 bytes are read sequentially and concatenated together. Then, a VAL function is performed on this string in order to determine the correct floating point value for 'A'. This same process is repeated for both the 'B' and the 'C' parameters with the exception that the 'B' parameter contains 5 bytes instead of 4.

Format of the Information Table Records. Each of the two information table records (WK and AR) has the same format, as follows:

20 rows
35 columns

Since it is possible to administer an item number larger than 256, the WinCast system uses a two byte scheme for each item number within the information table. The item number is determined as follows:

Assign a variable with 0;
add the value 256 to the variable when the second byte is larger than 0;
decrement this second byte and loop until the second byte becomes 0';
add the first byte to the variable.

The total number of entries in each information table is therefore: (rows * columns * 2).

CAST Maintenance Utility. Within the CAST development system there is a subdirectory within WinCast named 'Utility.' Inside this 'Utility' subdirectory exists a Visual Basic executable program named 'Project1.' This 'Project1' utility file can be used to add, delete, and edit the records and record keys of the CAST database file. Thus, this utility program can be used to add items or instructional text to the CAST 5 item banks, edit the text of existing instructions and question text, and delete selected text. It should be used only with great caution, however, because it is imperative that the key numbers of test questions, their item parameter values, and the item selection tables be coordinated. Failure to coordinate these properly could result in corruption of the adaptive test item selection and test scoring processes.

The EXAMINEE Database

The EXAMINEE database is designed to contain examinee identification information as well as all significant test performance information. The fields within this EXAMINEE database file are defined as follows:

Last name
First name
SSN (examinee)
Recruiter ID and password
Test information
Complete flag

Most of these fields are self explanatory with the exception of the 'Test Information' field and the 'Complete Flag' field. The field named 'Complete Flag' is set to false until the test administration for this record is completed. Upon the completion of this test session, this 'Complete flag' variable is set to TRUE. The Test Information field contains detailed test performance information including:

- I. Test instructions start time
- II. Test interruption time
- III. Test instructions completion time
- IV. Subtest label (Word Knowledge or Arithmetic Reasoning)
- V. Subtest start time
- VII. Item-by-item data, including:
 - VIII. Item ID number
 - IX. Item type (N = normal; E = experimental)
 - X. Item key (1 through 5)
 - XI. Item response (1 through 5)
 - XII. Item score (Y = right; N = wrong)
 - XIII. Updated ability estimate
 - XIV. Updated error variance
 - XV. Item response time in seconds
- XVI. Test results (basis for score report, including:
 - XVII. Predicted AFQT percentile score
 - XVIII. Predicted Word Knowledge standard score

XIX.	Predicted Arithmetic Reasoning standard score
XX.	AFQT range probability estimates
XXI.	50-99 (category 1 - 3a)
XXII.	31-49 (category 3b - 4a)
XXIII.	1-30 (category 4b - 5)

EXAMINEE database keys. There are no file keys defined in the EXAMINEE database. This is because a record is identified as one of the following:

- New record for a new test administration
- An active record that should be used to resume a previously started test.

The correct active record identified as one to be resumed is determined as follows:

- The 'Complete flag' is set to false
- The test was started on the same day
- A match is found on the Recruiter password
- A match is found on the examinee SSN

Since it is possible to have multiple records containing the same Recruiter password and examinee SSN, a simple key field (or concatenated fields used as a key) is not useful.

EXAMINEE database indexes. The EXAMINEE database uses two indexes in order to quickly search for specific records. These indexes are defined as follows:

- Recruiter password
- Examinee SSN

These two indexes help locate specific records when the system attempts to identify a resumed test.

The SECURITY Database

The SECURITY database is designed to contain recruiter passwords as well as an access level for each recruiter. In order to operate the WinCast system, the user must enter a recognized password. The password supplied by the user must be a 9-digit number that has been previously entered into the SECURITY database. Each recognized password has an associated level of access that was assigned when the user's password was initially registered.. This access level ranges from a value of 1 to 4. The values are intended to grant access to certain sensitive or potentially compromising CAST 5 software functions.

The fields defined within the SECURITY database are as follows:

- I. Recruiter password
- II. Recruiter name
- III. Access level

The RECRUITER PASSWORD field. The RECRUITER PASSWORD field is used to match the password entered by the user in order to determine if the user has a valid clearance to use the CAST system.

The name field. The name is only used for display purposes when modifying the SECURITY database. The user can see the recruiter's name in order to insure the proper maintenance functions are being performed on the correct recruiter password. The name is also recorded in the examinee record of tests administered by specific users.

The access level field. The Access level field is used to store a value from 1 to 4. The meaning of these four values are as follows:

- | | |
|----------------|--|
| Access level 1 | Can administer tests and generate score reports only |
| Access level 2 | Can administer level 1 user passwords |
| Access level 3 | Can administer level 2 user passwords, and use CAST 5 system maintenance utility functions |
| Access level 4 | System manager: no restrictions |

SECURITY database keys. There are no keys within the SECURITY database.

SECURITY database indexes. The SECURITY database uses the recruiter password field as its index. This is how the system can find a specific recruiter password within the SECURITY database.

SECURITY database maintenance. The WinCast system contains a utility that allows a user to add, update, and delete a recruiter's password information. This utility program allows the user to perform functions for other users who have an equal or lower Access level (with the exception of the Access level 4 user who can perform any function).

Bitmap Image Files

CAST 5 uses four bitmap image files as the source for graphics displays at various points in CAST test administration. Those files are listed here.

<u>File Name</u>	<u>Displays:</u>
CASTLOGN.BMP	The screen displayed at the start of the program.
ENDTEST.BMP	The screen displayed at test completion.
FEEDBACK.BMP	The score report screen.
PASSWORD.BMP	The screen displayed to elicit password entry.

References

- Knapp, D.J. (1987). *Final report on a national cross-validation of the Computerized Adaptive Screening Test (CAST)* (Technical Report 768). Alexandria VA: U.S. Army Research Institute for the Behavioral and Social Sciences.
- Knapp, D.J. & Pliske, R.M. (1986). *Preliminary report on a national cross-validation of the Computerized Adaptive Screening Test (CAST)* (Research Report 1430). Alexandria VA: U.S. Army Research Institute for the Behavioral and Social Sciences.
- Mathews, J.J., & Ree, M.J. (1982). *Enlistment Screening Test forms 81A and 81B: Development and calibration* (AFHRL Report No. 81-54). San Antonio, TX: Air Force Human Resources Laboratory.
- Park, R.K., & Dunn, M.L. (1996). *Compatibility evaluation and research on the Computerized Adaptive Screening test (CAST). Final report: User and programming guide*. Washington, DC: American Institutes for Research. (NTIS No. AD-A293 112).
- Sands, W.A., & Gade, P.A. (1983) An application of computerized adaptive testing in U.S. Army recruiting. *Journal of Computer-based Instruction*, 10, 87-89.
- Segall, D.O., Moreno, K.E., Bloxom, B.M., & Hetter, R.D. (in preparation). Psychometric procedures for administering CAT-ASVAB. In Sands, W.A., Waters, B.K., & McBride, J.R. (Eds). *Computerized adaptive testing: From inquiry to operation*. Chapter 12. Washington, D.C.: American Psychological Association.
- Wise, L.L., McHenry, J.J., Chia, W.J., Szenas, P.L., & McBride, J.R. (1989). *Refinement of the Computerized Adaptive Screening Test* (Final Report, Contract No. MDA903-86-C-0373). Washington, D.C.: American Institutes for Research.

Appendix A: CAST 5 for Windows User's Guide

Cast 5 for Windows

Users' Guide

Chapter 1 - Introduction to CAST 5.....	1
What is CAST?	1
Benefits of using CAST 5	1
Characteristics of CAST 5.....	1
What's different about CAST 5?	2
Chapter 2 - Getting Started Using CAST 5.....	3
Computer System Requirements	3
Computer Equipment	3
Operating System Software.....	3
Memory	3
Disk Storage Space.....	4
Installing CAST for Windows.....	4
Chapter 3 - Using CAST for Windows	5
Authorized users.....	5
Starting the CAST program.....	5
Chapter 4 - Administering and Scoring CAST Tests	7
Administering a CAST Test.....	7
If the Examinee Needs Help.....	9
If a CAST Test is Unintentionally Interrupted... ..	10
Getting CAST Test Scores	10
To End a CAST Session.....	12
Chapter 5 - Maintenance Functions.....	13
Levels of Access.....	13
Passwords and Password Maintenance	14
The File Utilities.....	15
View Records	16
Copy Records	18
Move Records	18
Delete Records	19

Introduction to CAST 5

What is CAST?

CAST is a computer software system designed to help recruiters make the best use of their own time and that of their recruiting prospects. CAST, which stands for "Computerized Adaptive Screening Test," is a short, computer-administered test that recruiters can use to forecast a prospect's likely performance on portions of the ASVAB.

Benefits of using CAST 5

CAST will predict a prospect's AFQT score as well as their scores on the Word Knowledge and Arithmetic Reasoning tests of the ASVAB. Recruiters can use these score forecasts to screen prospects, sorting those most likely to do well on the ASVAB from those who are not likely to qualify for enlistment or for specific enlistment programs.

Using computerized adaptive testing theory and technology, CAST does this very efficiently compared to alternative tests such as the Enlistment Screening Test (EST) or Computerized Enlistment Screening Test (CEST). A typical CAST session lasts less than 20 minutes -- far shorter than other screening tests. Despite its brevity, CAST is every bit as accurate as other tests for predicting AFQT scores. In fact, research has shown that CAST is as accurate as the AFQT itself.

Characteristics of CAST 5

CAST administers two short tests -- Word Knowledge and Arithmetic Reasoning. Both of these tests are very similar to the ASVAB tests of the same names. Both tests are adaptive, which means the computer selects test questions one at a time on the basis of the prospect's performance. Different people get different tests; adaptive selection of test questions ensures that time is not wasted with questions that are far too difficult or easy for the individual prospect's ability level.

The CAST 5 Word Knowledge test administers 15 questions; this test typically takes less than 5 minutes. CAST 5's Arithmetic Reasoning test administers between 7 and 12 questions, depending on the prospect's performance. It typically takes less than 15 minutes.

What's different about CAST 5?

CAST 5 is an improvement over previous versions of CAST. It is more accurate than previous editions for forecasting AFQT scores. Furthermore, it also provides forecasts of ASVAB Word Knowledge and Arithmetic Reasoning standard scores, something that earlier versions of CAST did not do.

CAST 5 is a Windows application. Its graphical user interface looks more attractive than older versions of CAST, is much easier to use, and includes some features that were not available previously in CAST. Because CAST 5 uses Windows, prospects can use the computer's "mouse" to answer the test questions; alternatively, they can use the standard keyboard if they prefer.

CAST 5's score reports are easier to interpret than those of previous editions. In addition, they are attractively formatted and suitable for inclusion in a prospect's recruiting package.

Getting Started Using CAST 5

Computer System Requirements

This section lists the computer equipment, software, memory and disk storage space needed to install and use CAST on your computer.

Computer Equipment

CAST 5 for Windows was designed for use on a standalone computer. It requires the following:

- An IBM PC compatible computer with a Pentium or superior processor
- A VGA or superior color video graphics adapter
- A keyboard as well as a mouse, trackball, or compatible pointing device
- A 1.44 MB 3.5-inch floppy disk drive

Operating System Software

CAST 5 requires a Microsoft Windows operating environment. While it was designed for use under Windows NT (version 3.5 or later), it is also compatible with Windows 95.

Memory

CAST 5 requires a minimum of 8 megabytes of random access memory.

Disk Storage Space

The CAST 5 software itself takes up less than 2 megabytes of disk storage space. However, at installation CAST 5 also installs certain required Windows system files if they are not already present on the computer's hard disk drive. In such cases, CAST 5 may require as much as 10 megabytes of disk storage.

Installing CAST for Windows

CAST 5 comes on a set of three 1.44 megabyte, 3.5-inch "Setup" disks. To install CAST 5, simply run the program SETUP.EXE contained on Setup disk number 1. From Windows 95 and Windows NT version 4, the procedure is as follows:

1. Start Windows NT or Windows 95.
2. Insert Setup Disk 1 in floppy disk drive a: or b:.
3. Click the Windows "Start" button, then choose "Run" from the menu that appears.
4. In the "Run" text box, type "a:setup" (or "b:setup"), then click "OK".
5. Follow the instructions on the screen. When a message asks which directory to use for CAST for Windows, enter (or confirm) "C:\WINCAST". On Windows NT systems, you may need to reserve memory; to do so, click the reserve memory option box that appears on the setup screen.

Using CAST for Windows

Authorized users

Once you have installed CAST 5 to directory C:\WINCAST, it can be used to administer the Computerized Adaptive Screening Test and to perform other CAST system functions. CAST 5 use, however, is limited to authorized users. CAST identifies authorized users by means of passwords; you must have a recognized password in order to use CAST.

At the completion of its installation, CAST 5 recognizes only one password -- that of the system manager. The system manager can use CAST's password maintenance function to create additional authorized passwords. The system manager may assign a single password, and authorize two or more recruiters to use it. Alternatively, the system manager may assign a different password to each authorized user. In any event, to receive a password to use CAST 5, contact your local system manager. The system manager must enter your assigned password on each computer you plan to use to administer CAST.

Note that CAST 5 has several different levels of access. Each authorized password is assigned a specific level of access, from Level 1 to Level 4. Individuals with Level 1 or higher access can administer CAST tests and generate score reports; this is all that most CAST users will require. Individuals with higher levels of access can perform certain maintenance and file utility functions. These functions, and the required access levels, are described in Chapter 5.

Starting the CAST program

Once you have received an authorized password, and the system manager has entered it into the computer you will use, you are ready to run the CAST 5 program.

To start using CAST 5, double-click the "CAST" button in the Windows 95/NT Programs menu. This will load the CAST 5 program; you will see the CAST menu screen, with the following message near the bottom:

"Type your password. Then press Enter."

Type your authorized password; click "OK" or press "Enter" to complete password entry. When you have entered an authorized password, you will see the main menu screen, which is depicted in Figure 1.

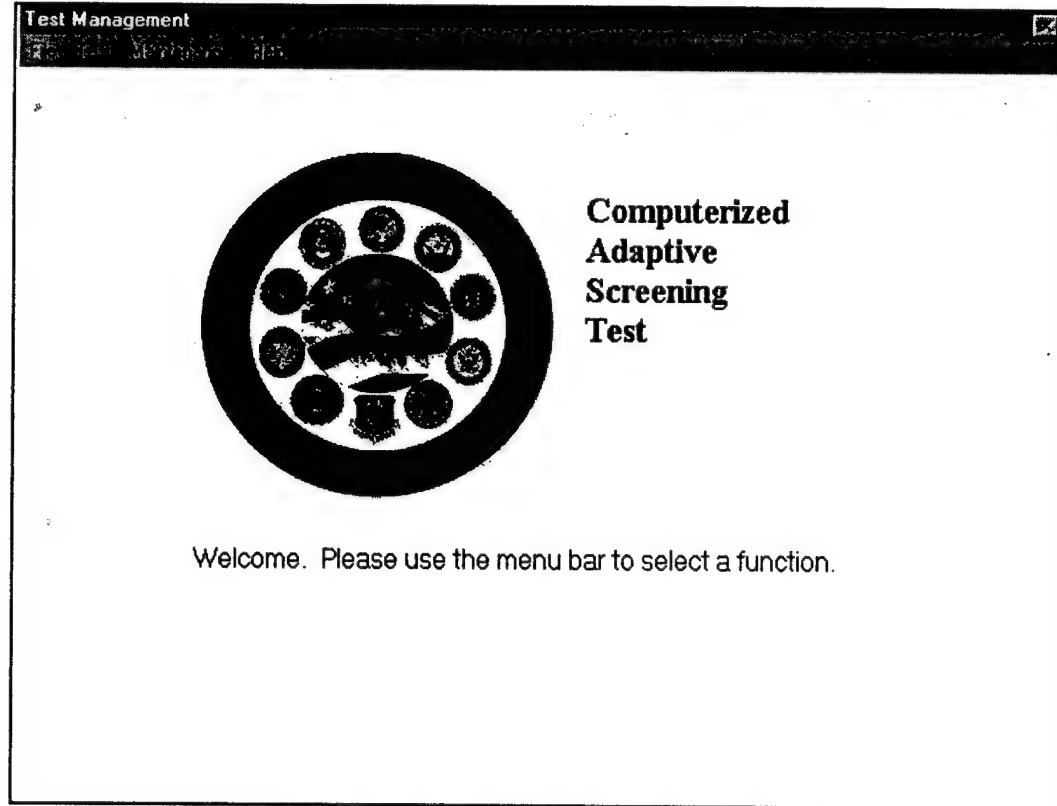


Figure 1. The CAST 5 Main Menu Screen.

As illustrated in Figure 1, the main menu screen has four submenus, listed in its menu bar. These are as follows:

1. File
2. Test
3. Maintenance
4. Help

The "Test" menu is all you need to give tests and display the results afterward. All four of the menu bar functions are described in Chapters 4 and 5.

Administering and Scoring CAST 5

The principal purpose of the CAST 5 software is to administer tests and inform the recruiter of the results. This chapter deals with both of these functions, along with some related features of the software.

Administering a CAST Test

To administer a CAST test, select "Test" from the main menu. Then select "Give a test", the first choice on the drop-down menu shown in Figure 2. You can use either the mouse or the keyboard to make your selection.

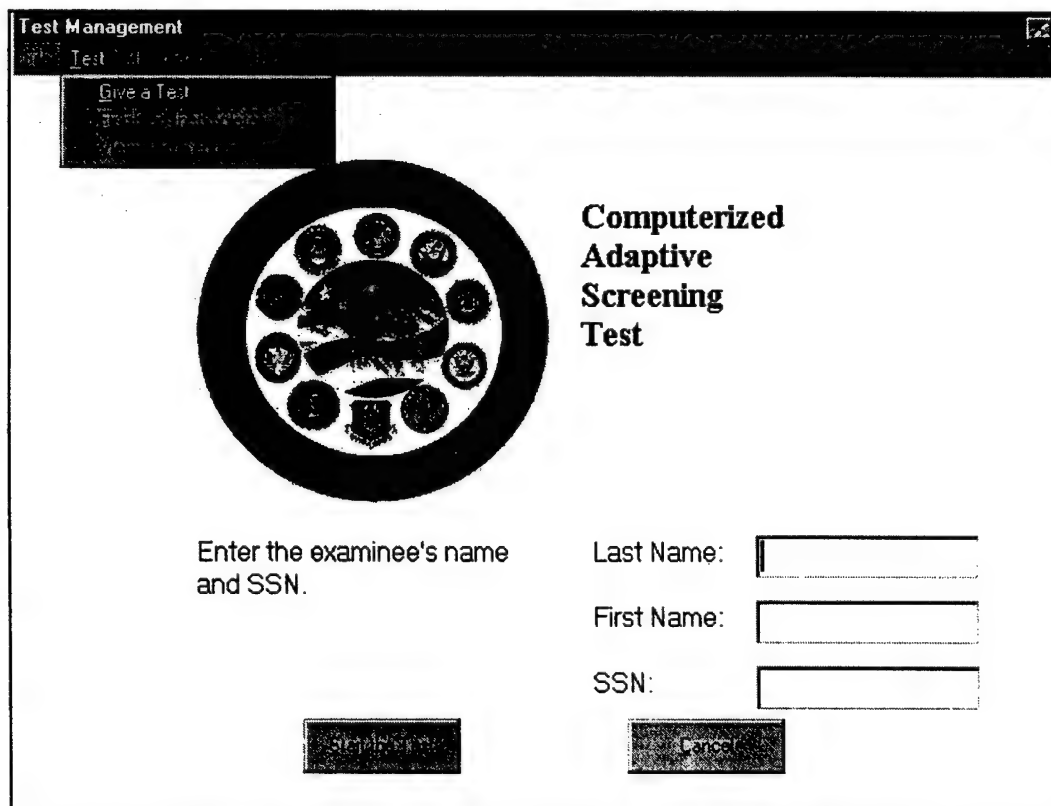
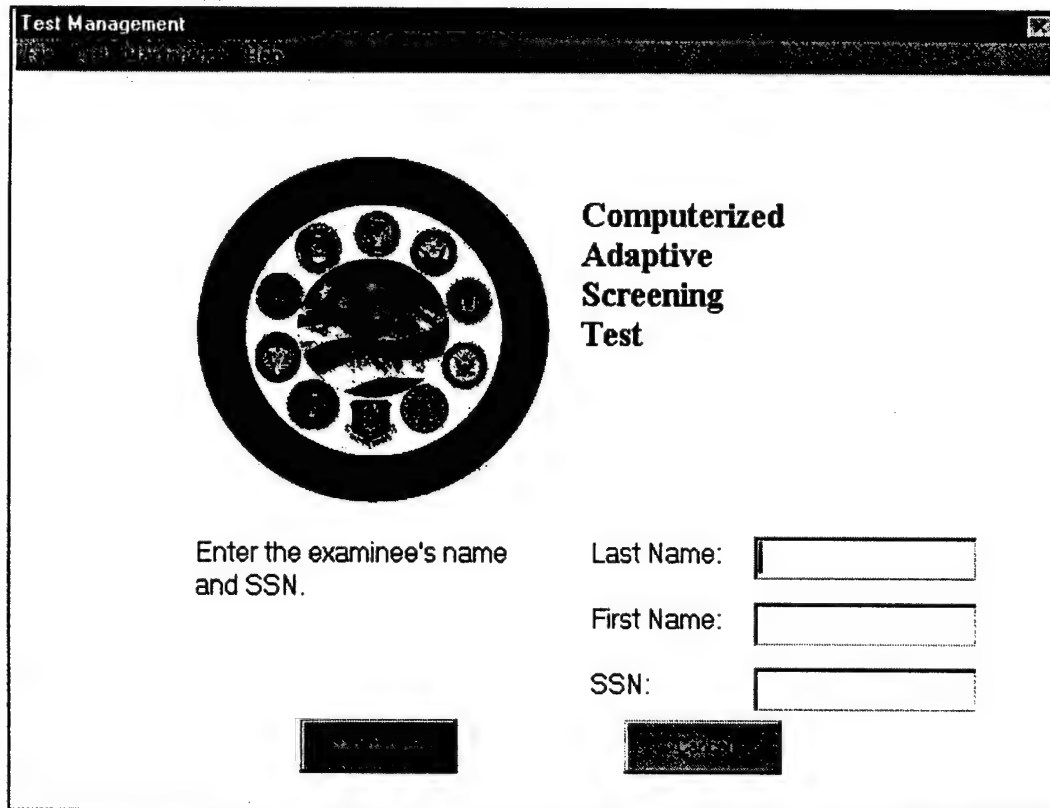


Figure 2. Menu Selections to Administer a CAST 5 Test.

As soon as you select "Give a test", an examinee identification screen like the one in Figure 3 will appear. Type the examinee's last name, first name, and social security number in the appropriate boxes. When you have finished, press Enter or click "Start the Test" to begin the test. Note that the test will not start until you have completely identified the examinee, including a valid 9-digit social security number.



Test Management

Computerized
Adaptive
Screening
Test

Enter the examinee's name
and SSN.

Last Name:

First Name:

SSN:

Start the Test Cancel

Figure 3. The Examinee Identification Entry Screen.

The CAST test will begin with a short set of orientation and general instruction screens. When they have finished, the CAST Word Knowledge test will be given; it will be followed by the Arithmetic Reasoning test. Each of these tests is preceded by an example question; when the examinee has answered the example question correctly, the test will be administered. The Word Knowledge test consists of 15 questions, and usually takes less than 5 minutes. The number of Arithmetic Reasoning test questions varies: It may have as few as 7 questions or as many as 12; it usually takes less than 15 minutes, but may take as much as 25 minutes for some examinees. After the first 7 Arithmetic Reasoning questions, the CAST software may give from 1 to 5 additional questions if needed to increase the accuracy of CAST's AFQT prediction.

All of the CAST test questions are in standard multiple choice format. Each answer choice is labeled "A", "B", "C", "D" or "E". Examinees must select the best of the answer choices presented, and must mark their answer before proceeding to the next question. Examinees may use the computer's mouse or trackball to select their answer choices; alternatively, they may use keys on the keyboard to do the same thing. To use the keyboard, they may press the A, B, C, D or E keys, or they may use the Tab key and the arrow keys to cycle among the answer choices.

There is no way to skip a question in CAST; examinees must answer every question. Once an answer has been marked, examinees must click "Next" or press the Enter key to proceed to the next question. They may change their answer at any time before doing this. However, once "Next" or "Enter" has been pressed, the answer can no longer be changed. CAST does not allow examinees to review previous questions, or to change their answers once a question has been finished.

If the Examinee Needs Help...

There is a "Help" button displayed at the bottom right of the screen during the test. Examinees can press "Help" to view a short summary of the test instructions. When they are finished with this, they can resume taking the test, or request help from the recruiter using a button labeled "Call the recruiter."

If an examinee opts for "Call the recruiter", a short message intended for the recruiter will appear on the screen. At this point, the recruiter has two options: 1) to return to the test, or 2) to stop the test. Choosing "Return to the test" will do just that. Choosing "Stop the test" will cause the test to be discontinued, and return to the CAST menu system; to prevent examinees from access to that system, password entry is required.

Stopping a CAST test does not have to end the examinee's test session. CAST allows an incomplete test to be resumed and finished, as long as it takes place on the same day as the interruption. The test will be resumed at the point of interruption; for example, if it was stopped at the fifth question, it will resume with question 5.

To resume an interrupted test, select "Finish an incomplete test" from the "Test" menu. When an incomplete test is resumed, CAST's general instructions will be repeated, followed by the example question for the interrupted test, and then the test itself. Stopping a test and then resuming it later is a convenient way to let an examinee who is having difficulty review all of the test directions.

If a CAST Test is Unintentionally Interrupted...

Sometimes a computer-administered test may be interrupted by accident. This might happen in the event of an electrical power failure, a battery failure, or an act on the examinee's part. If this occurs, the CAST test can be resumed at the point of the interruption, so that recruiter and examinee time are not wasted.

To resume an interrupted test, select "Finish an Incomplete Test" from the "Test" menu. CAST software will display a list of all that day's examinees who have incomplete tests. Select the examinee of interest, click "OK", and his/her test will resume immediately.

This feature only works for tests interrupted on the same day. A test cannot be resumed after the day it was begun.

Getting CAST Test Scores

CAST predicts examinees' scores on certain ASVAB tests: The AFQT percentile score, and standard scores for ASVAB's Word Knowledge and Arithmetic Reasoning tests. As soon as an examinee has completed the CAST tests, the screen directs them to tell their recruiter they are done. At this point, the test scores are available to the recruiter. Figure 4 depicts the screen that appears at the end of the test. When this screen appears, the recruiter can display or print out the examinee's score report immediately after entering an authorized password.

Figure 5¹ shows a sample CAST score report. In addition to the imaginary examinee's name and SSN, it lists CAST's prediction of the examinee's AFQT percentile score, and of his or her "Standard Scores" on ASVAB's Word Knowledge and Arithmetic Reasoning tests. In parentheses, the report also indicates how many Word Knowledge and Arithmetic Reasoning questions were given, and how long it took the examinee to complete each part of the test.

CAST saves test results in a data base on each CAST computer. To produce a score report for a previous examinee, select "Print" from the "File" menu. CAST will display a list of all examinees in the data base who finished their CAST tests. Once an examinee has been selected, their scores can be displayed on screen (by using the "Print preview" option) or sent to the printer.

¹ Figures 5 through 7 contain false names and social security numbers.



**Computerized
Adaptive
Screening
Test**

You have finished the test.

Please tell the test administrator you are done.

**Recruiter: To prepare a score report,
please enter your password.**



Figure 4. Enter a Password to Prepare a Score Report.

There is another way to determine CAST results. The "View test scores" option on the "Test" menu lists CAST results in a summary form for everyone in the data base who completed the CAST tests. The list is sorted by last name. It includes examinees' names, test administration dates, and predicted ASVAB scores; to aid in identification, it also includes the last 4 digits of their social security numbers.



Computerized Adaptive Screening Test

Score report for: Test, Example
000000000

Report date: 07-14-1997

Predicted AFQT Score 61

Word Knowledge	61 (15 items taken. Time: 01' 15")
Arithmetic Reasoning	65 (08 items taken. Time: 01' 19")

Note: The predicted AFQT is an estimated percentile, based on CAST validation research.

Word Knowledge and Arithmetic Reasoning are ASVAB Standard Scores, and are approximations. They are provided for recruiter convenience only; their accuracy has not been validated.



Figure 5. An Example CAST 5 Score Report.

To End a CAST Session

To terminate a CAST session, select "Exit" from the "File" menu.

Chapter 5

Maintenance Functions

Selecting "Maintenance Functions" from CAST's main menu displays a drop-down menu with two choices: "Passwords" and "File utilities". Use the "Password" menu to add or delete authorized CAST users, or to change their passwords. Use the "File utilities" menu to view test records in the CAST data base, or to copy, move or delete them. Each of these two functional areas is described more fully in the rest of this chapter. Preceding that is the following discussion of CAST's four levels of access.

Levels of Access

CAST's features and functions are available only to users with recognized passwords. To use CAST to administer tests or display test scores, you must enter any authorized password. To use any other CAST features, you must be authorized Level 2 or higher access. The following table summarizes CAST's four levels of access, and the system functions associated with each one. A check in the box under a specific access level indicates that a given CAST system function is available to users at that level; if the box is not checked, a higher level of access is required.

CAST System Function	Access Level			
	1	2	3	4
Administer CAST tests	✓	✓	✓	✓
Display and print individual examinees' scores	✓	✓	✓	✓
View test score lists	✓	✓	✓	✓
Add new authorized users to the CAST system		✓	✓	✓
Change authorized users' passwords		✓	✓	✓
Delete authorized users' passwords		✓	✓	✓
Use the file utilities			✓	✓

Note that while any authorized user can administer CAST tests, Level 2 or higher access is needed to add or delete authorized users, or to change passwords. In addition, Level 3 or higher access is required for use of the file utilities (which are described below).

In addition to the restrictions indicated above, Level 2 and 3 users are limited to administering lower-level password authorizations. That is, a Level 2 user may only add, change, and delete Level 1 users' passwords; Level 3 users may perform the same functions only for Level 1 and Level 2 users. In contrast, a Level 4 user may add, change, and delete passwords of any level, including Level 4, with one exception: The Level 4 System Manager can never be deleted.

Passwords and Password Maintenance

As mentioned above, CAST 5 cannot be used without first entering an authorized password. Installing CAST 5 by means of the SETUP program creates a single user with a Level 4 authorized password: the system manager. For security reasons, the system manager password is not printed in this User Guide; it will be transmitted separately to authorized CAST users at each location.

CAST 5 passwords may be any combination of letters and numbers; there are no restrictions on password length. The designers of CAST 5 intended for the system manager at each location to assign passwords to those with a need to administer CAST tests and perform other functions. In most cases, CAST 5 authorized users should be assigned Level 1 access only, unless there is a specific requirement for higher-level access.

Each time a CAST test is given, CAST 5 records the authorized user's password, and the associated user's name. If CAST system managers want to track CAST usage back to individual recruiters, they should assign each authorized user a unique password.

If individual accountability for CAST use is not required, system managers may prefer a less stringent system. For example, passwords such as "Army", "Navy", "Air Force", "Marines", and "Coast Guard" might be assigned for the purpose of simply accounting for CAST use by Service. Another alternative might be to use passwords to identify the recruiter's unit. Note that the software will not permit the same password to be assigned to more than one user, so a password assigned to an individual recruiter cannot be assigned to another recruiter using the same computer. CAST 5 will not allow more than one user to have the password "Army", for example.

Passwords can only be assigned, changed, or deleted by CAST users with Level 2 or higher access. A Level 1 user cannot create a new user, and cannot change or delete passwords.

CAST users with Level 2 and Level 3 access can perform all password maintenance functions. They are, however, restricted to password maintenance for users with lower-level access; they cannot create or modify passwords of their peers. For example, a Level 2 user can add Level 1 users and administer Level 1 passwords, but cannot add, delete or change Level 2 or Level 3 passwords.

Users authorized Level 4 access are not restricted in this way. A Level 4 user can create additional Level 4 users, and can change or delete Level 4 users' passwords. The only exception to this is the system manager password: It may be changed, but not deleted.

The File Utilities

CAST 5 includes several file utility functions. These have been designed primarily to support transmittal of CAST test records from each local computer to a central database; they may also be useful for local "housekeeping" purposes.

Each time a CAST test is given, a record of it is stored in a database file on the local computer. These records are intended for administrative and research use. For those purposes, it may be necessary to transmit the data to a centralized CAST data base in another location; the file utilities provide a means of doing this. Over time, CAST test records on a local computer can accumulate, taking up needed space on the local computer's hard disk drive; the file utilities provide a means of deleting these records. The file utilities also provide a means to inspect individual CAST test records at a fine level of detail; this may be useful for administrative, software maintenance, or research purposes.

Inspecting CAST test records, transmitting them to a central facility and deleting records are functions that must be performed with care and judgment. To reduce the opportunity for error or misuse, it is prudent to limit these functions to a small number of users. For this reason, the file utilities functions are reserved for users with the highest levels of access: Level 3 and Level 4. Users with Level 1 or 2 access cannot perform them.

Authorized users may access the file utilities by selecting the "File utilities" option under "Maintenance" on the main menu bar. When this is done, a screen like the one in Figure 6 will appear. It contains a list of all test records in the database; the test records are sorted by date, with the oldest

records listed first. It also displays command buttons for four functions: "Copy records", "Move records", "Delete records", and "View records".

Each of these functions is described in its own section below. Before using any of these functions, the user must first highlight the test records that are to be operated on. The descriptions of these operations begins with the "View records" function, and then proceeds to the other three functions.

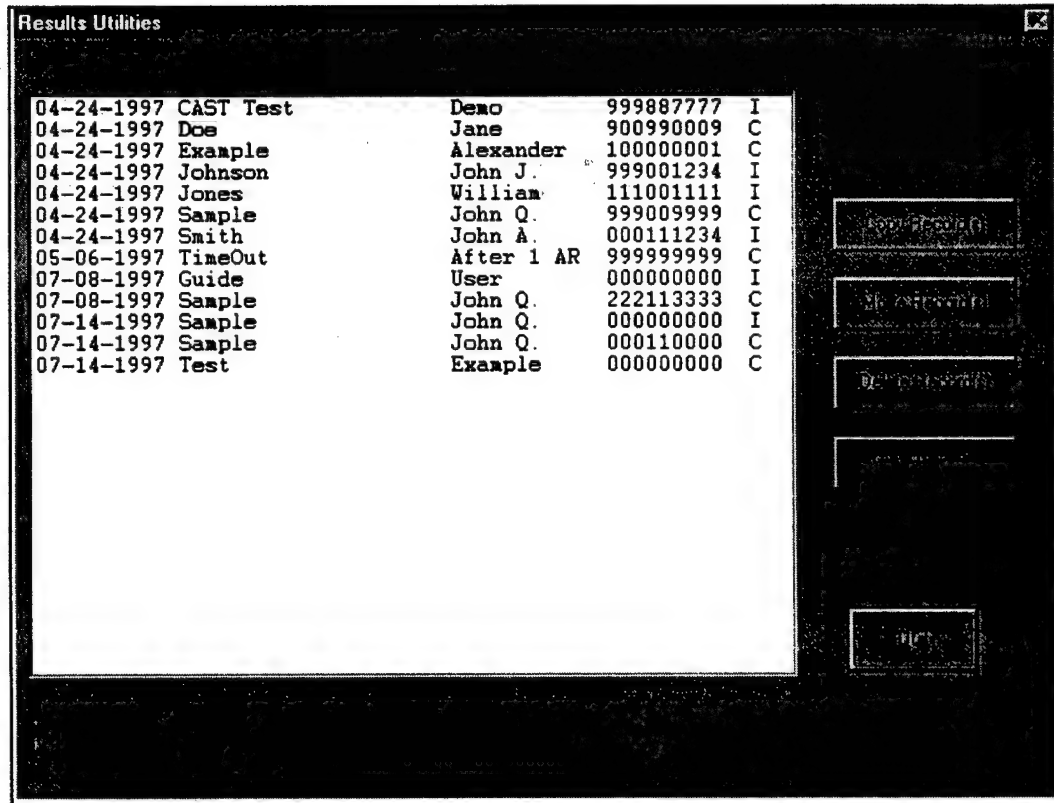


Figure 6. An Example of the File Utilities Control Screen.

View Records

"View records" allows authorized users to inspect the detailed records of each CAST test, an example of which is depicted in Figure 7. The CAST test record includes the examinee's name and social security number, the name and password used by the recruiter to initiate the test, the test date, and detailed records of the examinee's progress through the test directions and each Word Knowledge and Arithmetic Reasoning test question. There is a CAST test record for every test that was started, regardless of whether it was finished; for finished tests, the last entry in the record is a list of predicted ASVAB performance data, including the forecasts of ASVAB AFQT, Word Knowledge, and Arithmetic Reasoning scores. "View records" may be useful

for checking the accuracy of CAST's internal computations or ASVAB score predictions; it may also be useful for tracking CAST usage by Services or by individual recruiters.

View Examinee Record

Last Name: Test
 First Name: Example
 SSN: 00000000
 Recruiter ID: System Manager
 111223333
 Instruction Sequence Begin: 07-14-1997 14:30:45
 Instruction Sequence End: 07-14-1997 14:30:50
 WORD KNOWLEDGE:
 07-14-1997 14:30:52
 168 N 2 2 Y 000.4060 000.8463 0011
 220 N 3 3 Y 000.9355 000.5935 0003
 247 N 3 1 N 000.5078 000.3790 0004
 225 N 4 4 Y 000.8029 000.2984 0008
 237 N 1 2 N 000.4769 000.1868 0007
 227 N 3 3 Y 000.6827 000.1554 0004
 242 N 4 1 N 000.5271 000.1066 0003
 229 N 5 5 Y 000.6511 000.0944 0005
 241 N 4 1 N 000.5636 000.0789 0005
 226 N 4 4 Y 000.6523 000.0708 0003
 239 N 4 1 N 000.5794 000.0613 0004
 234 N 1 1 Y 000.6255 000.0596 0006
 236 N 1 2 N 000.5559 000.0527 0004
 231 N 1 1 Y 000.6193 000.0497 0004
 243 N 4 1 N 000.5827 000.0470 0004
 ARITHMETIC REASONING:

Figure 7. An Example of a CAST 5 Examinee Data base Record.

To use "View records", begin by highlighting the listed records of interest; then click on the "View records" button. The records will be displayed, one at a time, until all highlighted records have been inspected. To view the entire record, the user can use the up and down arrows, or the Page Up and Page Down keys to scroll forward and backward.

The first three lines of each CAST test record contain the examinee's last name, first name, and social security number. The next two lines contain data that identify the authorized user who initiated the test. Following that are the detailed records of the CAST test. These include beginning and ending times of the test instructions and of the tests themselves, and records of each test question. For each test question, the CAST record indicates the question identifier (e.g., 168N), its correct answer, the answer given, the item score ("Y" for right, "N" for wrong), two elements of ability estimation data, and the time in seconds taken to answer the question.

If the CAST test was completed, the very last line in the record contains six elements: The first one is the predicted AFQT percentile² score. The second and third are the predicted standard scores on ASVAB's Word Knowledge and Arithmetic Reasoning tests, respectively; these score predictions use the ASVAB standard score³ scale. The fourth, fifth and sixth elements are the computed probabilities that the examinee's AFQT score will fall into AFQT categories 1 through 3a (AFQT 50 - 99), 3b (AFQT 31 - 49), and 4 through 5 (AFQT 1 - 30).

Copy Records

The "Copy records" button causes highlighted examinees' records to be copied from the CAST 5 test records database to another database file (called the "target file" below.) The target file must be a database file with the same format as CAST's test records database file. It is intended to be used to facilitate transmitting CAST test records from the local computer to a central CAST database: Copy the records to the target file, which can then be transmitted elsewhere by a variety of means, including but not limited to modem transfers, e-mail, and even floppy disks sent by surface mail.

To copy CAST test records, the user must highlight the records to be copied, then click the "Copy records" button. A screen message will ask the user to specify the name of the target file -- the one that is to receive the data. Note that the target file must be a data base file in the same format as CAST's test records data base file. CAST provides an empty formatted file upon installation. That file's name is "copydata.mdb"; users should make a copy of that file, with a different file name, for use with the "Copy records" function⁴.

Move Records

The "Move records" function performs the same function as "Copy records". However, once the selected records have been copied, using the "Move" function, they are deleted from the CAST examinee record file.

² ASVAB AFQT scores are reported as "percentiles": the proportion of equal or lower scores in a reference group, the 1980 Profile of American Youth population.

³ The ASVAB standard score scale is used to record scores on the 10 ASVAB tests. 50 is average; the scale has a standard deviation of 10. Scores below 30 and above 70 are rare.

⁴ This point is important: Do not move or copy examinee records to the "MOVEDATA.MDB" file. First, make a copy of "MOVEDATA.MDB" with a different filename prefix (retain the ".MDB" suffix.) Always use a file copy to move or copy examinee records. This is necessary because there is no provision to delete data copied or moved to the target file.

The "Move records" function is intended to facilitate transferring selected examinee records from local computers to a central CAST database. For example, a group of records can be "moved" from the examinee file to a target file; then the target file could be uploaded to the central database, by mail, modem, or Internet server.

As mentioned above under "Copy records", the target file must be a data base file in the same format as CAST's test records data base file. CAST provides an empty formatted file named "COPYDATA.MDB"; users should make a copy of that file, with a different file name, for use with the "Copy records" or "Move records" functions.

Since "Move records" deletes the designated records from the CAST examinee file, it can simplify file housekeeping.

Delete Records

"Delete records" permanently deletes selected examinees' records from the CAST computer's examinee data file. This function may be useful for trimming the examinee file of unwanted data. However, deletion destroys the affected data, and should be used with great caution. If CAST examinee records are to be aggregated at a central database, "Move records" or "Copy records" should be used, rather than "Delete records."

Appendix B: CAST 5 Source Code

Appendix B
CAST 5 Source Code

Table of Contents

CONTROL.BAS	2
ABOUT.FRM	10
DIALOG.FRM	12
EXAMUTIL.FRM	17
FEEDBACK.FRM	27
FILESELECT.FRM	33
FORM1.FRM	37
HELP.FRM	38
ITEMWINDOW.FRM	40
PWORD.FRM	70
SAMPLES.FRM	79
SECURITY.FRM	82
VIEWEXAMINEE.FRM	102

CONTROL.BAS

Attribute VB_Name = "ControlMod"
Option Explicit

'All significant system variables are defined below

' (32 bit code) Declare Function SystemParametersInfo Lib "user32" Alias
"SystemParametersInfoA" (ByVal uAction As Long, ByVal uParam As Long, ByVal
lpvParam As Any, ByVal fuWinIni As Long) As Long
Declare Function SystemParametersInfo Lib "user" (ByVal uAction As Integer,
ByVal uParam As Integer, lpvParam As Any, ByVal fuWinIni As Integer) As
Integer

Public Const SPIF_SENDWININICHANGE = &H2
Public Const SPI_SETDESKWALLPAPER = 20
Public Const SPI_SETSCREENSAVEACTIVE = 17

Global Abort As Integer
Global InfoRows As Integer
Global InfoColumns As Integer
Global First As String
Global Last As String
Global SSN As String
Global Version As String
Global VersionDate As String

Global Const MaxItems = 300
Global Const MaxExpItems = 50
Global Const MaxDistractors = 5
Global Const MaxSubtests = 2
Global Const MaxIncorrect = 3
Global Const ParameterFileName = "C:\WINCAST\CAST.INI"
Global Const SecurityFileName = "C:\WINCAST\SECURITY.MDB"
Global Const Offset = 8
Global Const INTRO = 0
Global Const WK = 1
Global Const AR = 2
Global Const FINAL = 3
Global Const InverseOn = 162
Global Const InverseOff = 163
Global Const DefaultName = "System Manager"
Global Const HELPSCREEN = 1
Global Const QUIT = 2
Global Const ASK_ADMINISTRATOR = 3
Global Const ALLOW_BACKUP = 4

Global CRLF As String * 2

Global InitPVAR(MaxSubtests) As Double
Global InitTheta(MaxSubtests) As Double
Global ExtraItems(MaxSubtests) As Integer
Global ExtraValue(1 To 3) As Double
Global Info As String * 1400 'Room for Info Table
Global RNDItems(MaxDistractors) As Integer
Global Used(MaxItems) As Integer
Global ExpUsed(MaxItems) As Integer

Global DebugFlag As Integer
Global AllItemDebug As Integer
Global EnableShowStats As Integer
Global MaxWKItems As Integer
Global MaxARItems As Integer

```

Global WKLength As Integer
Global ARLength As Integer
Global ProbParams(1 To 6) As Double

Global RegCoefWK As Double
Global RegCoefAR As Double
Global RegConst As Double
Global WriteDisk As Integer

'Global TempFile As String
'Global OutputFile As String
'Global ReportProgram As String
'Global InputWork As String
'Global OutputWork As String

Global MaxWKExp As Integer
Global MaxARExp As Integer
Global WKExpAdmin As Integer
Global ARExpAdmin As Integer

Global WKExp(1 To MaxExpItems, 1 To MaxSubtests) As Integer
Global ARExp(1 To MaxExpItems, 1 To MaxSubtests) As Integer
Global WKTheta As Double
Global ARTheta As Double
Global WKPvar As Double
Global ARPvar As Double
'
'   New Vars
'
Global DataPath As String
Global IntroImage As String
Global DatabaseName As String
Global ExamineeDBName As String

Global FColor As Integer
Global BColor As Integer
Global DColor As Integer
Global HColor As Integer

Global CastDb As Database
Global CastTable As Table
Global ExamineeDb As Database
Global ExamineeTable As Table

Global SendString As String
Global SendInt As Integer
Global IntroScreenStart As Integer
Global IntroScreenEnd As Integer
Global Banner As String
Global BannerX, BannerY As Integer
Global ItemFontSize As Integer
Global GeneralErrorSwitch As Integer
Global RecruiterName As String
Global RecruiterSSN As String
Global Restart As Integer
Global OutputString As String
Global P1 As Double
Global P2 As Double
Global P3 As Double
Global AFQT As Double
Global AWK As Double
Global AAR As Double

```

```

Global WKPerformance As String * 50
Global ARPerformance As String * 50

Global AccessLevel As Integer
Global ARTimeLimit As Integer

Global DriveSave As String
Global DirSave As String

Private Sub ScreenSaveOff_Click()
'Turns the screen saver mechanism off
    Dim R As Integer
    R = SystemParametersInfo(SPI_SETSCREENSAVEACTIVE, 0&, 0&,
SPIF_SENDWININICHANGE)
End Sub

Private Sub ScreenSaveOn_Click()
'Turns the screen saver mechanism back on
    Dim R As Integer
    R = SystemParametersInfo(SPI_SETSCREENSAVEACTIVE, 1&, 0&,
SPIF_SENDWININICHANGE)
End Sub

Private Sub SetWallPaper_Click()
'Defines the background wallpaper
    Dim R As Integer
    R = SystemParametersInfo(SPI_SETDESKWALLPAPER, 0, "d:\\winnt\\ball.bmp",
SPIF_SENDWININICHANGE)
End Sub

Sub GetPerformance(TestInfo As String)
'This checks the examinee's performance for:
'Number of items taken and the number of minutes and seconds that have
'transpired during the test administration
    Dim EndFound As Integer
    Dim Temp As String
    Dim Result As String
    Dim TestNumber As Integer
    Dim Count(1 To 2) As Integer
    Dim TimeCount(1 To 2) As Integer
    Dim X As Integer
    Dim Min As Integer
    Dim Sec As Integer

    Count(1) = 0
    Count(2) = 0
    TimeCount(1) = 0
    TimeCount(2) = 0
    TestNumber = 1
    Temp = TestInfo
    EndFound = False
    While Not EndFound
        EndFound = ParseString(Temp, Result)
        If Mid$(Result, 4, 1) = " " And Mid$(Result, 6, 1) = " " Then
            If Mid$(Result, 5, 1) = "N" Then 'Not experimental
                Count(TestNumber) = Count(TestNumber) + 1
                TimeCount(TestNumber) = TimeCount(TestNumber) + Val(Mid$(Result,
33, 4))
            End If
        ElseIf InStr(1, Result, "ARITHMETIC REASONING:", 1) > 0 Then
            TestNumber = 2
        End If
    End Sub

```

```

Wend

For X = 1 To 2
    Sec = TimeCount(X)
    Min = 0
    While Sec > 59
        Min = Min + 1
        Sec = Sec - 60
    Wend
    If X = 1 Then
        WKPerformance = " ( " & Format$(Count(1), "00") & " items taken.
Time: " & Format$(Min, "00") & "' " & Format$(Sec, "00") & Chr$(34) & ")"
    Else
        ARPerformance = " ( " & Format$(Count(2), "00") & " items taken.
Time: " & Format$(Min, "00") & "' " & Format$(Sec, "00") & Chr$(34) & ")"
    End If
Next X
End Sub

```

```

Sub QSort(Array() As String, N As Integer)
'This is a generic sort utility that can be used anywhere in the program.
'At this time it is being used to view test scores.
    Dim LSTK(100) As Integer
    Dim RSTK(100) As Integer
    Dim Dummy As String
    Dim X As String
    Dim S As Integer
    Dim L As Integer
    Dim R As Integer
    Dim I As Integer
    Dim J As Integer

    S = 1
    LSTK(1) = 1
    RSTK(1) = N
10100:
    L = LSTK(S)
    R = RSTK(S)
    S = S - 1
10200:
    I = L
    J = R
    X = Array(Int((L + R) / 2))
10300:
    If Array(I) < X Then GoTo 10320
    GoTo 10340
10320:
    I = I + 1
    GoTo 10300
10340:
    If X < Array(J) Then GoTo 10360
    GoTo 10400
10360:
    J = J - 1
    GoTo 10340
10400:
    If I <= J Then GoTo 10420
    GoTo 10500
10420:
    Dummy = Array(I)

```

```

    Array(I) = Array(J)
    Array(J) = Dummy
    I = I + 1
    J = J - 1
10500:
    If I <= J Then GoTo 10300
    If I >= R Then GoTo 10650
    S = S + 1
    LSTK(S) = I
    RSTK(S) = R
10650:
    R = J
    If L < R Then GoTo 10200
    If S <> 0 Then GoTo 10100
End Sub

Sub SystemPause(Interval As Long)
'This is a system pause that can be called from anywhere. This is used
'in some places to slow the system down a little bit in order to insure
'the system does not jump ahead too quickly.
    Dim X As Long

    X = Timer
    While Timer - X < Interval
        DoEvents
    Wend
End Sub

Function ParseString(Temp As String, Result As String) As Integer
'This routine is used throughout the system to parse a string and
'return TRUE if the end of the string has been reached, or FALSE if
'more of the string needs to be parsed.
    Dim X As Integer

    X = InStr(Temp, Chr$(10))
    If X = 0 Then
        Result = Temp
        Temp = ""
        ParseString = True
        Exit Function
    Else
        Result = Left$(Temp, X - 2)
        Temp = Right(Temp, Len(Temp) - X)
        ParseString = False
    End If
End Function

Sub Init()
'This routine initializes all of the significant global system variable
    Dim FileNum, Z As Integer
    Dim Temp$
    Dim WKExpCount As Integer
    Dim ARExpCount As Integer
    Dim AccumExpItems As Integer
    Dim NoOfExp, Position As Integer

    CRLF = Chr$(13) & Chr$(10)
    Abort = False
    GeneralErrorSwitch = False
    Randomize Timer

```

```

AccessLevel = 0
WriteDisk = True

InitPVAR(1) = 1
InitPVAR(2) = 1
InitTheta(1) = 0
InitTheta(2) = 0
InfoRows = 20
InfoColumns = 35

FileNum = FreeFile
Open ParameterFileName For Input As #FileNum
On Local Error GoTo FileReadError
Line Input #FileNum, Temp$
Version = RTrim(LTrim(Temp$))
Line Input #FileNum, Temp$
VersionDate = RTrim(LTrim(Temp$))
Line Input #FileNum, Temp$
DatabaseName = RTrim(LTrim(Temp$))
Line Input #FileNum, Temp$
ExamineeDBName = RTrim(LTrim(Temp$))
Line Input #FileNum, Temp$
DataPath = RTrim(LTrim(Temp$))
Line Input #FileNum, Temp$
IntroImage = RTrim(LTrim(Temp$))

Line Input #FileNum, Temp$
FColor = Val(Temp$)
Line Input #FileNum, Temp$
BColor = Val(Temp$)
Line Input #FileNum, Temp$
DColor = Val(Temp$)
Line Input #FileNum, Temp$
HColor = Val(Temp$)

Line Input #FileNum, Temp$
IntroScreenStart = Val(Temp$)
Line Input #FileNum, Temp$
IntroScreenEnd = Val(Temp$)
Line Input #FileNum, Temp$
ItemFontSize = Val(Temp$)
Line Input #FileNum, Temp$
ARTimeLimit = Val(Temp$) * 60

Input #FileNum, DebugFlag, Temp$
Input #FileNum, AllItemDebug, Temp$
Input #FileNum, EnableShowStats, Temp$
Input #FileNum, MaxWKItems, Temp$
Input #FileNum, MaxARItems, Temp$
Input #FileNum, WKLength, Temp$
Input #FileNum, ARLength, Temp$
If AllItemDebug Then
    WKLength = MaxWKItems
    ARLength = MaxARItems
End If
Input #FileNum, ExtraItems(1), Temp$
Input #FileNum, ExtraValue(1), Temp$
Input #FileNum, ExtraItems(2), Temp$
Input #FileNum, ExtraValue(2), Temp$
Input #FileNum, ExtraValue(3), Temp$
For Z = 1 To 6
    Input #FileNum, ProbParams(Z): Input #FileNum, Temp$

```

Next Z

```
Input #FileNum, RegCoefWK, Temp$
Input #FileNum, RegCoefAR, Temp$
Input #FileNum, RegConst, Temp$
'Input #FileNum, WriteDisk, Temp$
'Input #FileNum, TempFile, Temp$
'Input #FileNum, OutputFile, Temp$
'Input #FileNum, ReportProgram, Temp$
'Input #FileNum, InputWork, Temp$
'Input #FileNum, OutputWork, Temp$
```

```
Input #FileNum, MaxWKExp, Temp$
Input #FileNum, MaxARExp, Temp$
Input #FileNum, WKExpAdmin, Temp$
Input #FileNum, ARExpAdmin, Temp$
```

WKExpCount = 0

If WKExpAdmin = 0 Then

 Input #FileNum, Temp\$, Temp\$

Else

 AccumExpItems = 0

 While AccumExpItems < WKExpAdmin

 Input #FileNum, NoOfExp, Position

 AccumExpItems = AccumExpItems + NoOfExp

 WKExpCount = WKExpCount + 1

 WKExp(WKExpCount, 1) = NoOfExp

 WKExp(WKExpCount, 2) = Position

 Wend

 Input #FileNum, Temp\$

 WKExp(WKExpCount + 1, 1) = 0

 WKExp(WKExpCount + 1, 2) = 0

End If

ARExpCount = 0

If ARExpAdmin = 0 Then

 Input #FileNum, Temp\$, Temp\$

Else

 AccumExpItems = 0

 While AccumExpItems < ARExpAdmin

 Input #FileNum, NoOfExp, Position

 AccumExpItems = AccumExpItems + NoOfExp

 ARExpCount = ARExpCount + 1

 ARExp(ARExpCount, 1) = NoOfExp

 ARExp(ARExpCount, 2) = Position

 Wend

 Input #FileNum, Temp\$

 ARExp(ARExpCount + 1, 1) = 0

 ARExp(ARExpCount + 1, 2) = 0

End If

Close #FileNum

ItemWindow.Cls

Set CastDb = OpenDatabase(DatabaseName)

Set CastTable = CastDb.OpenTable("ScreenText")

Set ExamineeDb = OpenDatabase(ExamineeDBName)

Set ExamineeTable = ExamineeDb.OpenTable("Examinee")

Exit Sub

NoParamsFound:

 MsgBox "Cannot find the parameter file: " & ParameterFileName

End

FileReadError:

```

    MsgBox "Having trouble reading file: " & ParameterFileName
    CastTable.Close
    ExamineeTable.Close
End
End Sub

Sub Main()
'This 'Main()' routine controls the flow of the main menu,
'the test administration, and the main menu again.
    Dim R As Integer

    Call ScreenSaveOff_Click
    'R = SystemParametersInfo(17, 0&, 0&, &H2)
    'MsgBox Str$(R)

    If Not DebugFlag Then
        On Error GoTo GeneralError
    End If
    Call Init
    While True
        Security.Show MODAL
        If SendString <> "Quit" Then
            ItemWindow.Show MODAL
        Else
            CastTable.Close
            ExamineeTable.Close
            Call ScreenSaveOn_Click
        End
    End If
Wend
End

GeneralError:
    GeneralErrorSwitch = True
    SendInt = 0
    Dialog.Show MODAL
End
End Sub

```

ABOUT.FRM

VERSION 4.00

Begin VB.Form About

```

AutoRedraw      = -1  'True
Caption         = "About Cast"
ClientHeight    = 4380
ClientLeft      = 2655
ClientTop       = 1905
ClientWidth     = 3870
Height          = 4785
Left            = 2595
LinkTopic       = "Form1"
MaxButton       = 0   'False
MinButton       = 0   'False
ScaleHeight     = 4380
ScaleWidth      = 3870
Top             = 1560
Width           = 3990

```

Begin VB.Frame Frame2

```

Height          = 735
Left            = 960
TabIndex        = 2
Top             = 3600
Width           = 2175

```

Begin VB.CommandButton Command1

```

Caption         = "&Continue"
Default         = -1  'True
Height          = 375
Left            = 360
TabIndex        = 3
Top             = 240
Width           = 1455

```

End

End

Begin VB.Frame Frame1

```

Height          = 3495
Left            = 240
TabIndex        = 0
Top             = 0
Width           = 3495

```

Begin VB.Label Label3

```

Alignment       = 2   'Center
BackColor       = &H00C0C0C0&
BeginProperty Font
    name         = "MS Sans Serif"
    charset      = 1
    weight       = 400
    size         = 12
    underline    = 0   'False
    italic       = 0   'False
    strikethrough = 0   'False
EndProperty
ForeColor       = &H00C00000&
Height          = 855
Left            = 240
TabIndex        = 5
Top             = 2280
Width           = 3015

```

```

End
Begin VB.Label Label2
    Alignment       = 2 'Center
    BackColor       = &H00C0C0C0&
    BeginProperty Font
        name        = "MS Sans Serif"
        charset      = 1
        weight       = 400
        size         = 12
        underline    = 0 'False
        italic       = 0 'False
        strikethrough = 0 'False
    EndProperty
    ForeColor       = &H00000040&
    Height          = 855
    Left            = 240
    TabIndex        = 4
    Top             = 1320
    Width           = 3015
End
Begin VB.Label Label1
    Alignment       = 2 'Center
    BackColor       = &H00C0C0C0&
    BeginProperty Font
        name        = "MS Sans Serif"
        charset      = 1
        weight       = 400
        size         = 12
        underline    = 0 'False
        italic       = 0 'False
        strikethrough = 0 'False
    EndProperty
    ForeColor       = &H000000FF&
    Height          = 855
    Left            = 240
    TabIndex        = 1
    Top             = 360
    Width           = 3015
End
End
Attribute VB_Name = "About"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Option Explicit

Private Sub Command1_Click()
    SendString = ""
    Unload About
End Sub

Private Sub Form_Load()
    Label1.Caption = "Cast for Windows"
    Label2.Caption = VersionDate
    Label3.Caption = "Version: " & Version
End Sub

```

DIALOG.FRM

VERSION 4.00

Begin VB.Form Dialog

```

AutoRedraw      = -1 'True
BackColor       = &H00C0C0C0&
BorderStyle     = 0 'None
ClientHeight    = 4230
ClientLeft     = 1080
ClientTop      = 1515
ClientWidth    = 6720
ClipControls   = 0 'False
ControlBox     = 0 'False
Height         = 4635
Left           = 1020
LinkTopic      = "Form1"
MaxButton      = 0 'False
MinButton      = 0 'False
NegotiateMenus = 0 'False
ScaleHeight    = 4230
ScaleWidth     = 6720
ShowInTaskbar  = 0 'False
Top            = 1170
Width          = 6840
WindowState    = 2 'Maximized

```

Begin VB.CommandButton Command3

```

Caption      = "&Back"
Height       = 495
Left         = 2760
TabIndex     = 6
Top          = 6000
Visible      = 0 'False
Width        = 1575

```

End

Begin VB.CommandButton Command1

```

Caption      = "&Next"
Height       = 495
Left         = 4200
TabIndex     = 5
Top          = 6000
Width        = 1575

```

End

Begin VB.CommandButton Command2

```

Caption      = "&Call the Recruiter"
Height       = 495
Left         = 5760
TabIndex     = 2
Top          = 6000
Visible      = 0 'False
Width        = 2055

```

End

Begin VB.Frame Frame1

```

Height      = 6975
Left        = 0
TabIndex    = 0
Top         = 0
Width       = 9615

```

Begin VB.PictureBox Picture1

```

Height      = 6615

```

```

Left           = 360
ScaleHeight    = 6555
ScaleWidth     = 8835
TabIndex       = 1
Top            = 120
Width          = 8895
Begin VB.Frame Frame2
    Caption     = "Password required: Type it and press ENTER."
    Height      = 855
    Left        = 4320
    TabIndex    = 3
    Top         = 5520
    Visible     = 0 'False
    Width       = 3615
    Begin VB.TextBox Text1
        Height      = 375
        Left        = 120
        PasswordChar = "*"
        TabIndex    = 4
        Text        = "Text1"
        Top         = 360
        Width       = 3375
    End
End
End
End
End
Attribute VB_Name = "Dialog"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Option Explicit

Dim Offset As Integer
Dim RecordStart As Integer
Dim RecordEnd As Integer
Dim Ptr As Integer
Dim LMargin As Integer
Dim MyDb As Database
Dim MyTableDef As Table
Dim Processing As Integer

Sub GeneralErrorMessage()
    'This is the system's generic error message.
    Dim Message As String
    Dim LMargin As Integer

    Message = "Warning:" & Chr$(10) & Chr$(10)
    Message = Message & Space$(LMargin) & "Test-taker:" & Chr$(10) & Chr$(10)
    Message = Message & Space$(LMargin) & "The computer program has encountered  

a problem." & Chr$(10)
    Message = Message & Space$(LMargin) & "Please tell the recruiter you need  

assistance." & Chr$(10) & Chr$(10) & Chr$(10) & Chr$(10)
    Message = Message & Space$(LMargin) & "Recruiter:" & Chr$(10) & Chr$(10)
    Message = Message & Space$(LMargin) & "The computer program has detected a  

problem." & Chr$(10)
    Message = Message & Space$(LMargin) & "You will not be able to complete the  

current test." & Chr$(10)
    Message = Message & Space$(LMargin) & "Try to administer the test again.  

If this error recurs," & Chr$(10)
    Message = Message & Space$(LMargin) & "notify your system administrator of  

the problem."

```

```

    Dialog.Cls
    Dialog.CurrentX = 4200
    Dialog.CurrentY = 500
    Dialog.Print Message
End Sub

Sub ShowScreen(ScreenNumber As Integer)
'This routine displays a specific record from the CAST database file
'on to the screen.
    If ScreenNumber = -1 Then
        Call GeneralErrorMessage
        Exit Sub
    End If

    CastTable.Index = "Primary"
    CastTable.Seek "=", ScreenNumber
    If Not CastTable.NoMatch Then
        Picture1.Cls
        Picture1.Print CastTable("TextInfo")
    End If

    If Banner <> "" Then
        Picture1.CurrentX = BannerX
        Picture1.CurrentY = BannerY
        Picture1.Print Banner
    End If
End Sub

Function VerifyPassword() As Integer
'This routine checks to make sure that the SSN (password) entered
'is found in the SECURITY database.
    Set MyDb = OpenDatabase(SecurityFileName)
    Set MyTableDef = MyDb.OpenTable("Security")
    MyTableDef.Index = "SSNIndex"
    MyTableDef.Seek "=", LTrim(RTrim(Text1.Text))
    If MyTableDef.NoMatch Then
        VerifyPassword = False
    Else
        VerifyPassword = True
    End If
    MyTableDef.Close
End Function

Private Sub Command1_Click()
    If Processing Then
        Exit Sub
    End If

    Ptr = Ptr + 1
    If Ptr <= RecordEnd Then
        Call ShowScreen(Ptr)
        Command1.Left = 5280
        Command3.Visible = True
        Processing = True
        Call SystemPause(0.5)
        Processing = False
    Else
        SendInt = 0
        Unload Dialog
    End If
End Sub

```

```

Private Sub Command2_Click()
    If Processing Then
        Exit Sub
    End If
    If SendInt = HELPSCREEN Then
        SendInt = ASK_ADMINISTRATOR
        Command2.Caption = "Stop the test"
        Command1.SetFocus
        Call ShowScreen(200)
    ElseIf SendInt = ASK_ADMINISTRATOR Then
        'In order to require a password, please remove comments and last 2 lines
        'Frame2.Visible = True
        'Text1.Text = ""
        'Text1.Visible = True
        'Text1.SetFocus
        'Command2.Visible = False
        SendInt = QUIT
        Unload Dialog
    End If
End Sub

Private Sub Command3_Click()
    If Processing Then
        Exit Sub
    End If

    Ptr = Ptr - 1
    If Ptr < RecordStart Then
        If ALLOW_BACKUP Then
            Unload Dialog
            Exit Sub
        Else
            Ptr = RecordStart
        End If
    End If
    Call ShowScreen(Ptr)
    Processing = True
    Call SystemPause(0.5)
    Processing = False
    If Ptr = RecordStart And SendInt <> ALLOW_BACKUP Then
        Command1.Left = 4200
        Command3.Visible = False
    End If
    Command1.SetFocus
End Sub

Private Sub Form_Activate()
    LMargin = 10
    Processing = False
    If GeneralErrorSwitch Then
        Picture1.FontSize = 12
        Call ShowScreen(-1)
    Else
        Picture1.FontSize = ItemFontSize
        Ptr = RecordStart
        Call ShowScreen(Ptr)
    End If
    Command1.SetFocus
End Sub

```

```

Private Sub Form_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        Call Command1_Click
    End If
End Sub

Private Sub Form_Load()
    Dim X As Integer

    If SendInt = HELPSCREEN Then
        Command1.Caption = "&Return to the Test"
        Command1.Left = 1800
        Command2.Visible = True
    End If
    X = InStr(SendString, ",")
    Offset = Val(Left$(SendString, X - 1)) * 1000
    SendString = Right$(SendString, Len(SendString) - X)
    X = InStr(SendString, ",")
    RecordStart = Val(Left$(SendString, X - 1)) + Offset
    SendString = Right$(SendString, Len(SendString) - X)
    RecordEnd = Val(SendString) + Offset
    If SendInt = ALLOW_BACKUP Then
        Command1.Left = 5280
        Command3.Visible = True
    End If
End Sub

Private Sub Picture1_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        Call Command1_Click
    End If
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        If VerifyPassword() Then
            SendInt = QUIT
            Unload Dialog
        Else
            MsgBox "This password is not recognized."
            Text1.Text = ""
        End If
    End If
End Sub

```

EXAMUTIL.FRM

VERSION 4.00

Begin VB.Form ExamUtil

```

Caption           = "Results Utilities"
ClientHeight      = 4230
ClientLeft        = 1440
ClientTop         = 1425
ClientWidth       = 6720
ClipControls      = 0 'False
ControlBox        = 0 'False
Height            = 4635
Left              = 1380
LinkTopic         = "Form1"
MaxButton         = 0 'False
MinButton         = 0 'False
ScaleHeight       = 4230
ScaleWidth        = 6720
Top               = 1080
Width             = 6840
WindowState       = 2 'Maximized

```

Begin VB.CommandButton Command7

```

Caption           = "&View Record(s)"
Height            = 495
Left              = 7560
TabIndex          = 9
Top               = 3600
Width             = 1695

```

End

Begin VB.CommandButton Command6

```

Cancel            = -1 'True
Caption           = "&Cancel"
Height            = 495
Left              = 7440
TabIndex          = 8
Top               = 6000
Visible           = 0 'False
Width             = 1695

```

End

Begin VB.CommandButton Command5

```

Caption           = "&Delete Record(s)"
Height            = 495
Left              = 7560
TabIndex          = 7
Top               = 2880
Width             = 1695

```

End

Begin VB.CommandButton Command4

```

Caption           = "&Move Record(s)"
Height            = 495
Left              = 7560
TabIndex          = 6
Top               = 2160
Width             = 1695

```

End

Begin VB.CommandButton Command3

```

Caption           = "&Copy Record(s)"
Height            = 495

```

```

        Left           = 7560
        TabIndex       = 5
        Top            = 1440
        Width          = 1695
    End

    Begin VB.DirListBox Dir1
        Height          = 1605
        Left            = 7320
        TabIndex        = 4
        Top             = 1560
        Width           = 2175
    End

    Begin VB.DriveListBox Drive1
        Height          = 315
        Left            = 7320
        TabIndex        = 3
        Top             = 1200
        Width           = 2175
    End

    Begin VB.FileListBox File1
        Height          = 1815
        Left            = 7440
        Pattern          = "*.mdb"
        TabIndex        = 2
        Top             = 3240
        Width           = 1815
    End

    Begin VB.ListBox List2
        BeginProperty Font
            name          = "Courier"
            charset       = 1
            weight        = 400
            size          = 9.75
            underline     = 0 'False
            italic        = 0 'False
            strikethrough  = 0 'False
        EndProperty
        Height          = 5325
        Left            = 120
        MultiSelect      = 2 'Extended
        TabIndex        = 1
        Top             = 480
        Width           = 7095
    End

    Begin VB.CommandButton Command1
        Caption          = "&Quit"
        Height           = 615
        Left             = 7680
        TabIndex         = 0
        Top              = 5160
        Width            = 1215
    End

End

Attribute VB_Name = "ExamUtil"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Option Explicit

```

```

Const MaxElement = 500
Dim PArray(MaxElement) As Integer

Dim LastName As String
Dim FirstName As String
Dim RecruiterSSN As String
Dim TestInfo As String
Dim CompleteFlag As Integer
Dim SSN As String
Dim DestFileName As String

Dim DestDb As Database
Dim DestTable As Table

Sub GetRecord(RecordNumber As Integer)
'This routine gets a specific record number (identified by RecordNumber)
'and buffers the fields into system variables.
    Dim X As Integer

    ExamineeTable.Index = "SSNPrimary"
    ExamineeTable.MoveFirst
    For X = 0 To RecordNumber - 2
        ExamineeTable.MoveNext
    Next X
    LastName = ExamineeTable("LastName")
    FirstName = ExamineeTable("FirstName")
    RecruiterSSN = ExamineeTable("RecruiterSSN")
    TestInfo = ExamineeTable("TestInfo")
    CompleteFlag = ExamineeTable("CompleteFlag")
    SSN = ExamineeTable("SSN")
End Sub

Function Pad(Chars As String, Length As Integer) As String
'This routine will verify that a string has exactly the number
'of characters specified by the parameter 'Length'.
    Chars = LTrim(RTrim(Chars))
    If Len(Chars) > Length Then
        Chars = Left$(Chars, Length)
    ElseIf Len(Chars) <> Length Then
        While Len(Chars) < Length
            Chars = Chars & " "
        Wend
    End If
    Pad = Chars
End Function

Sub CopyRecord()
'This routine copies a record from the EXAMINEE database to an
'external database. Prior to this, the external database must
'have been formatted in the same exact way the EXAMINEE database
'is formatted.

```

```

Dim Temp As String
Dim X As Integer
Dim Count As Integer

If Not OpenDestFile Then
    Exit Sub
End If

Count = 0
For X = 0 To List2.ListCount - 1
    If List2.Selected(X) Then
        Call GetRecord(PArray(X + 1))
        DestTable.AddNew
        DestTable("LastName") = LastName
        DestTable("FirstName") = FirstName
        DestTable("RecruiterSSN") = RecruiterSSN
        DestTable("TestInfo") = TestInfo
        DestTable("CompleteFlag") = CompleteFlag
        DestTable("SSN") = SSN
        DestTable.Update
        Count = Count + 1
    End If
Next X
MsgBox Str$(Count) & " record(s) copied"
DestTable.Close
End Sub

Sub DeleteRecord()
    'This routine removes records from the EXAMINEE database.
    Dim Temp As String
    Dim X As Integer
    Dim Y As Integer
    Dim Count As Integer

    Count = 0
    For X = 0 To List2.ListCount - 1
        If List2.Selected(X) Then
            Call GetRecord(PArray(X + 1))
            ExamineeTable.Delete
            Count = Count + 1
            For Y = 0 To List2.ListCount - 1
                If List2.Selected(Y) Then
                    If PArray(Y + 1) > PArray(X + 1) Then
                        PArray(Y + 1) = PArray(Y + 1) - 1
                    End If
                End If
            Next Y
        End If
    Next X
    ExamineeTable.Close
    Set ExamineeDb = OpenDatabase(ExamineeDBName)
    Set ExamineeTable = ExamineeDb.OpenTable("Examinee")
    Call ReadFile
    MsgBox Str$(Count) & " record(s) deleted"
End Sub

Sub MoveRecord()
    'This routine moves (copies then deletes original) records from the EXAMINEE
    database to an

```

'external database. Prior to this, the external database must
'have been formatted in the same exact way the EXAMINEE database
'is formatted.

```
Dim Temp As String
Dim X As Integer
Dim Y As Integer
Dim Count As Integer
```

```
If Not OpenDestFile Then
    Exit Sub
End If
```

```
Count = 0
'ExamineeTable.Index = "SSNPrimary"
'ExamineeTable.MoveFirst
For X = 0 To List2.ListCount - 1
    If List2.Selected(X) Then
        Call GetRecord(PArray(X + 1))
        'LastName = ExamineeTable("LastName")
        'FirstName = ExamineeTable("FirstName")
        'RecruiterSSN = ExamineeTable("RecruiterSSN")
        'TestInfo = ExamineeTable("TestInfo")
        'CompleteFlag = ExamineeTable("CompleteFlag")
        'SSN = ExamineeTable("SSN")
```

```
DestTable.AddNew
DestTable("LastName") = LastName
DestTable("FirstName") = FirstName
DestTable("RecruiterSSN") = RecruiterSSN
DestTable("TestInfo") = TestInfo
DestTable("CompleteFlag") = CompleteFlag
DestTable("SSN") = SSN
DestTable.Update
ExamineeTable.Delete
Count = Count + 1
For Y = 0 To List2.ListCount - 1
    If List2.Selected(Y) Then
        If PArray(Y + 1) > PArray(X + 1) Then
            PArray(Y + 1) = PArray(Y + 1) - 1
        End If
    End If
Next Y
End If
```

```
Next X
'ExamineeTable.MoveNext
Next X
MsgBox Str$(Count) & " record(s) moved"
DestTable.Close
ExamineeTable.Close
Set ExamineeDb = OpenDatabase(ExamineeDBName)
Set ExamineeTable = ExamineeDb.OpenTable("Examinee")
Call ReadFile
End Sub
```

Function OpenDestFile() As Integer
'This routine simply opens an external file formatted in the same
'way the EXAMINEE database is formatted.

On Local Error GoTo BadDb

```
Set DestDb = OpenDatabase(DestFileName)
Set DestTable = DestDb.OpenTable("Examinee")
OpenDestFile = True
Exit Function
```

```

BadDb:
    MsgBox "Target file is not a valid examinee formatted file."
    OpenDestFile = False
End Function

Sub ReadFile()
'The EXAMINEE database is read and the records are placed into the
'List Box.
    Dim Token As String * 1
    Dim Temp As String
    Dim Status As String
    Dim Array(MaxElement) As String
    Dim Point As Integer
    Dim Y As Integer
    Dim X As Integer
    Dim TestInfo As String
    Dim TestDate As String
    Dim Sequence As Integer

    Token = Chr$(0)
    List2.Clear
    Point = 0
    Sequence = 0
    On Local Error GoTo FileEmpty

    ExamineeTable.Index = "SSNPrimary"
    ExamineeTable.MoveFirst
    While Not ExamineeTable.EOF
        TestInfo = ExamineeTable("TestInfo")
        X = InStr(1, TestInfo, "Instruction Sequence Begin:", 1)
        If X = 0 Then
            TestDate = ""
        Else
            TestDate = Mid$(TestInfo, X + 27, 10)
        End If
        If ExamineeTable("CompleteFlag") = True Then
            Status = "C"
        Else
            Status = "I"
        End If
        Temp = TestDate & " " & Pad(ExamineeTable("LastName"), 20) & " "
        Temp = Temp & Pad(ExamineeTable("FirstName"), 12)
        Temp = Temp & Pad(ExamineeTable("SSN"), 10) & " "
        Temp = Temp & Pad(Status, 1)
        Sequence = Sequence + 1
        Temp = Temp & Token & Str$(Sequence)
        If Point < MaxElement Then
            Point = Point + 1
            Array(Point) = Temp
        End If
        ExamineeTable.MoveNext
    Wend
    Call QSort(Array(), Point)
    For Y = 1 To Point
        X = InStr(1, Array(Y), Token, 1)
        If X <> 0 Then
            PArray(Y) = Val(Right$(Array(Y), Len(Array(Y)) - X))
            List2.AddItem Left$(Array(Y), X - 1)
        Else
            PArray(Y) = 1
            List2.AddItem Array(Y)
        End If
    Next Y
End Sub

```

```

        End If
    Next Y
    ExamineeTable.MoveFirst
    Exit Sub

FileEmpty:
    Exit Sub
End Sub

Sub ViewRecord()
'This routine will display one examinee record.
    Dim Temp As String
    Dim X As Integer
    Dim Y As Integer

    List2.Enabled = False
    For X = 0 To List2.ListCount - 1
        If List2.Selected(X) Then
            Call GetRecord(PArray(X + 1))
            SendString = "Last Name: " & LastName & CRLF
            SendString = SendString & "First Name: " & FirstName & CRLF
            SendString = SendString & "SSN: " & SSN & CRLF
            SendString = SendString & "Recruiter ID: "
            Y = InStr(TestInfo, Chr$(10))
            If Y <> 0 Then
                TestInfo = Right$(TestInfo, Len(TestInfo) - Y)
            End If
            SendString = SendString & TestInfo
            ViewExaminee.Show MODAL
        End If
    Next X
    List2.Enabled = True
End Sub

Private Sub Command1_Click()
    SendString = ""
    Unload ExamUtil
End Sub

Private Sub Command3_Click()
    Dim X As Integer
    Dim Found As Integer

    If List2.ListCount < 1 Then
        MsgBox "There are no examinee records to copy."
        Exit Sub
    End If
    For X = 0 To List2.ListCount - 1
        If List2.Selected(X) Then
            Found = True
            X = List2.ListCount
        End If
    Next X
    If Not Found Then
        MsgBox "You must select examinee record(s) from the list."
        Exit Sub
    End If
    FileSelect.Show MODAL
    If SendString <> "" Then
        If SendString = ExamineeDBName Then

```

```

        MsgBox "You cannot select the examinee database currently in use."
        Exit Sub
    End If
    DestFileName = SendString
    'If Not OpenDestFile Then
    '    Exit Sub
    'End If
    Call CopyRecord
End If
End Sub

Private Sub Command4_Click()
    Dim X As Integer
    Dim Found As Integer

    If List2.ListCount < 1 Then
        MsgBox "There are no examinee records to copy."
        Exit Sub
    End If
    For X = 0 To List2.ListCount - 1
        If List2.Selected(X) Then
            Found = True
            X = List2.ListCount
        End If
    Next X
    If Not Found Then
        MsgBox "You must select examinee record(s) from the list."
        Exit Sub
    End If
    FileSelect.Show MODAL
    If SendString <> "" Then
        If SendString = ExamineeDBName Then
            MsgBox "You cannot select the examinee database currently in use."
            Exit Sub
        End If
        DestFileName = SendString
        'If Not OpenDestFile Then
        '    exit sub
        ' endif
        Call MoveRecord
    End If
End Sub

Private Sub Command5_Click()
    Dim X As Integer
    Dim Found As Integer

    If List2.ListCount < 1 Then
        MsgBox "There are no examinee records to delete."
        Exit Sub
    End If

    Found = False
    For X = 0 To List2.ListCount - 1
        If List2.Selected(X) Then
            Found = True
            X = List2.ListCount
        End If
    Next X
    If Not Found Then
        MsgBox "You must select examinee record(s) from the list."
        Exit Sub
    End If

```

```

    End If
    If List2.ListCount < 1 Then
        MsgBox "There are no examinee records to delete."
        Exit Sub
    End If
    Call DeleteRecord
End Sub

Private Sub Command6_Click()
    Command1.Visible = True
    Command3.Visible = True
    'Command2.Visible = True
    Command4.Visible = True
    Command5.Visible = True
    Command6.Visible = False
    Command7.Visible = True
    Dir1.Visible = False
    Drive1.Visible = False
    File1.Visible = False
End Sub

Private Sub Command7_Click()
    Dim X As Integer
    Dim Found As Integer

    If List2.ListCount < 1 Then
        MsgBox "There are no examinee records to view."
        Exit Sub
    End If

    Found = False
    For X = 0 To List2.ListCount - 1
        If List2.Selected(X) Then
            Found = True
            X = List2.ListCount
        End If
    Next X
    If Not Found Then
        MsgBox "You must select examinee record(s) from the list."
        Exit Sub
    End If
    Call ViewRecord
End Sub

Private Sub Dir1_Change()
    File1.Path = Dir1.Path
End Sub

Private Sub Drive1_Change()
    Dir1.Path = Drive1.Drive
End Sub

Private Sub Form_Load()
    DriveSave = ""
    DirSave = ""
    Dir1.Visible = False
    Drive1.Visible = False
    File1.Visible = False
    Call ReadFile
End Sub

```


FEEDBACK.FRM

VERSION 4.00

Begin VB.Form Feedback

ClientHeight = 4230
ClientLeft = 1095
ClientTop = 1515
ClientWidth = 6720
ClipControls = 0 'False
ControlBox = 0 'False

BeginProperty Font

name = "Times New Roman"
charset = 1
weight = 400
size = 12
underline = 0 'False
italic = 0 'False
striketrough = 0 'False

EndProperty

ForeColor = &H00C00000&
Height = 4635
Left = 1035
LinkTopic = "Form1"
MaxButton = 0 'False
MinButton = 0 'False
NegotiateMenus = 0 'False
ScaleHeight = 4230
ScaleWidth = 6720
Top = 1170
Width = 6840
WindowState = 2 'Maximized

Begin VB.CommandButton Command2

Caption = "&Print"
Default = -1 'True

BeginProperty Font

name = "MS Sans Serif"
charset = 1
weight = 400
size = 8.25
underline = 0 'False
italic = 0 'False
striketrough = 0 'False

EndProperty

Height = 495
Left = 7680
TabIndex = 8
Top = 5880
Visible = 0 'False
Width = 1695

End

Begin VB.TextBox Text1

Height = 495
Left = 6480
PasswordChar = "*"
TabIndex = 1
Text = "Text1"
Top = 5880
Visible = 0 'False
Width = 2895

End

```

Begin VB.CommandButton Command1
  Caption      = "&OK"
  BeginProperty Font
    name        = "MS Sans Serif"
    charset     = 1
    weight      = 400
    size        = 8.25
    underline   = 0 'False
    italic      = 0 'False
    strikethrough = 0 'False
  EndProperty
  Height       = 495
  Left         = 7680
  TabIndex     = 0
  Top          = 6480
  Width        = 1695
End

```

```

Begin VB.Label Label9
  BackColor    = &H00FFFFFF&
  BeginProperty Font
    name        = "Arial"
    charset     = 1
    weight      = 700
    size        = 9.75
    underline   = 0 'False
    italic      = 0 'False
    strikethrough = 0 'False
  EndProperty
  ForeColor    = &H00C00000&
  Height       = 255
  Left         = 3000
  TabIndex     = 11
  Top          = 3960
  Visible      = 0 'False
  Width        = 3015
End

```

```

Begin VB.Label Label8
  BackColor    = &H00FFFFFF&
  BeginProperty Font
    name        = "Arial"
    charset     = 1
    weight      = 700
    size        = 9.75
    underline   = 0 'False
    italic      = 0 'False
    strikethrough = 0 'False
  EndProperty
  ForeColor    = &H00C00000&
  Height       = 375
  Left         = 3840
  TabIndex     = 10
  Top          = 5040
  Visible      = 0 'False
  Width        = 5415
End

```

```

Begin VB.Label Label7
  BackColor    = &H00FFFFFF&
  BeginProperty Font
    name        = "Arial"

```

```

        charset      = 1
        weight       = 700
        size         = 9.75
        underline    = 0 'False
        italic       = 0 'False
        strikethrough = 0 'False
    EndProperty
    ForeColor      = &H00C00000&
    Height         = 375
    Left           = 3840
    TabIndex       = 9
    Top            = 4800
    Visible        = 0 'False
    Width          = 5415
End

```

```

Begin VB.Label Label6
    BackColor      = &H00FFFFFF&
    BeginProperty Font
        name        = "Arial"
        charset     = 1
        weight      = 700
        size        = 9.75
        underline    = 0 'False
        italic      = 0 'False
        strikethrough = 0 'False
    EndProperty
    ForeColor      = &H00C00000&
    Height         = 255
    Left           = 4680
    TabIndex       = 7
    Top            = 6480
    Visible        = 0 'False
    Width          = 1335
End

```

```

Begin VB.Label Label5
    BackColor      = &H00FFFFFF&
    BeginProperty Font
        name        = "Arial"
        charset     = 1
        weight      = 700
        size        = 9.75
        underline    = 0 'False
        italic      = 0 'False
        strikethrough = 0 'False
    EndProperty
    ForeColor      = &H00C00000&
    Height         = 255
    Left           = 4680
    TabIndex       = 6
    Top            = 6240
    Visible        = 0 'False
    Width          = 1335
End

```

```

Begin VB.Label Label4
    BackColor      = &H00FFFFFF&
    BeginProperty Font
        name        = "Arial"
        charset     = 1
        weight      = 700
    EndProperty

```

```

        size           = 9.75
        underline      = 0   'False
        italic         = 0   'False
        strikethrough  = 0   'False
    EndProperty
    ForeColor          = &H00C00000&
    Height             = 255
    Left               = 4680
    TabIndex           = 5
    Top                = 6000
    Visible            = 0   'False
    Width              = 1335
End

```

```

Begin VB.Label Label3
    BackColor          = &H00FFFFFF&
    BeginProperty Font
        name           = "Arial"
        charset        = 1
        weight         = 700
        size           = 9.75
        underline      = 0   'False
        italic         = 0   'False
        strikethrough  = 0   'False
    EndProperty
    ForeColor          = &H00C00000&
    Height             = 375
    Left               = 7560
    TabIndex           = 4
    Top                = 3720
    Visible            = 0   'False
    Width              = 1695
End

```

```

Begin VB.Label Label2
    BackColor          = &H00FFFFFF&
    BeginProperty Font
        name           = "Arial"
        charset        = 1
        weight         = 700
        size           = 9.75
        underline      = 0   'False
        italic         = 0   'False
        strikethrough  = 0   'False
    EndProperty
    ForeColor          = &H00C00000&
    Height             = 375
    Left               = 3840
    TabIndex           = 3
    Top                = 4320
    Visible            = 0   'False
    Width              = 2175
End

```

```

Begin VB.Label Label1
    BackColor          = &H00FFFFFF&
    BeginProperty Font
        name           = "Arial"
        charset        = 1
        weight         = 700
        size           = 9.75
        underline      = 0   'False
    EndProperty

```

```

        italic          = 0   'False
        strikethrough   = 0   'False
    EndProperty
    ForeColor           = &H00C00000&
    Height              = 255
    Left                = 3000
    TabIndex            = 2
    Top                 = 3720
    Visible             = 0   'False
    Width               = 3015
End

Begin VB.Image Image1
    Height              = 6015
    Left                = 240
    Top                 = 240
    Width               = 9135
End
End
Attribute VB_Name = "Feedback"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Option Explicit

Dim MyDb As Database
Dim MyTableDef As Table

Private Sub Command1_Click()
    If SendString = "1" Then
        If LTrim(RTrim(Text1.Text)) = "" Then
            MsgBox "Recruiter: You must enter your password to continue."
            Exit Sub
        End If

        Set MyDb = OpenDatabase(SecurityFileName)
        Set MyTableDef = MyDb.OpenTable("Security")
        MyTableDef.Index = "SSNIndex"
        MyTableDef.Seek "=", LTrim(RTrim(Text1.Text))
        If MyTableDef.NoMatch Then
            MsgBox "Cannot Find: " & Text1.Text & ". Please try again."
            Exit Sub
        End If
        MyTableDef.Close
    End If
    Unload Feedback
End Sub

Private Sub Command2_Click()
    Command1.Visible = False
    Command2.Visible = False
    Feedback.PrintForm
    Command1.Visible = True
    Command2.Visible = True
End Sub

Private Sub Form_Activate()
    If SendString = "1" Then
        Text1.SetFocus
    Else
        Label1.Caption = Last & ", " & First
    End If
End Sub

```

```

Label1.Visible = True

Label2.Caption = Format$(AFQT, "00")
Label2.Visible = True
Label3.Caption = Date$
Label3.Visible = True

Label7.Caption = Format$(AWK, "00") & " " & WKPerformance
Label7.Visible = True
Label8.Caption = Format$(AAR, "00") & " " & ARPerformance
Label8.Visible = True
Label9.Caption = SSN
Label9.Visible = True
If SendString = "3" Then
    Command1.Visible = False
    Command2.Visible = False
    Feedback.PrintForm
    DoEvents
    Unload Feedback
Else
    Command2.Visible = True
End If
End If
End Sub

Private Sub Form_Load()
    If SendString = "1" Then
        Feedback.Picture = LoadPicture(DataPath & "endtest.bmp")
        Text1.Text = ""
        Text1.Visible = True
    Else
        Feedback.Picture = LoadPicture(DataPath & "feedback.bmp")
        Text1.Visible = False
    End If
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        Call Command1_Click
    End If
End Sub

```

FILESELECT.FRM

VERSION 4.00

Begin VB.Form FileSelect

```

BackColor      = &H00C0C0C0&
Caption        = "Select Database File"
ClientHeight   = 3195
ClientLeft     = 1140
ClientTop      = 1515
ClientWidth    = 6690
Height         = 3600
Left           = 1080
LinkTopic      = "Form1"
ScaleHeight    = 3195
ScaleWidth     = 6690
Top            = 1170
Width          = 6810

```

Begin VB.CommandButton Command2

```

Caption        = "&Cancel"
Height         = 375
Left           = 5280
TabIndex       = 5
Top            = 1440
Width          = 1335

```

End

Begin VB.CommandButton Command1

```

Caption        = "&OK"
Height         = 375
Left           = 5280
TabIndex       = 4
Top            = 720
Width          = 1335

```

End

Begin VB.TextBox Text1

```

Height         = 375
Left           = 240
TabIndex       = 3
Top            = 720
Width          = 1935

```

End

Begin VB.FileListBox File1

```

Height         = 1815
Left           = 240
TabIndex       = 2
Top            = 1200
Width          = 1935

```

End

Begin VB.DirListBox Dir1

```

Height         = 1605
Left           = 2520
TabIndex       = 1
Top            = 600
Width          = 2175

```

End

Begin VB.DriveListBox Drive1

```

Height         = 315
Left           = 2520

```

```

        TabIndex      = 0
        Top           = 2760
        Width         = 2175
    End

    Begin VB.Label Label3
        Caption        = "Drives"
        BeginProperty Font
            name         = "MS Sans Serif"
            charset      = 1
            weight       = 700
            size         = 8.25
            underline    = 0 'False
            italic       = 0 'False
            strikethrough = 0 'False
        EndProperty
        Height         = 375
        Left           = 2520
        TabIndex      = 8
        Top           = 2400
        Width         = 1695
    End

    Begin VB.Label Label2
        Caption        = "Folders"
        BeginProperty Font
            name         = "MS Sans Serif"
            charset      = 1
            weight       = 700
            size         = 8.25
            underline    = 0 'False
            italic       = 0 'False
            strikethrough = 0 'False
        EndProperty
        Height         = 375
        Left           = 2520
        TabIndex      = 7
        Top           = 120
        Width         = 1695
    End

    Begin VB.Label Label1
        Caption        = "File Name"
        BeginProperty Font
            name         = "MS Sans Serif"
            charset      = 1
            weight       = 700
            size         = 8.25
            underline    = 0 'False
            italic       = 0 'False
            strikethrough = 0 'False
        EndProperty
        Height         = 375
        Left           = 240
        TabIndex      = 6
        Top           = 240
        Width         = 1695
    End

    End

    Attribute VB_Name = "FileSelect"
    Attribute VB_Creatable = False
    Attribute VB_Exposed = False

```

Option Explicit

```
Private Sub Command1_Click()  
    DriveSave = Drive1.Drive  
    DirSave = Dir1.Path  
    SendString = Text1.Text  
    Unload FileSelect  
End Sub
```

```
Private Sub Command2_Click()  
    DriveSave = Drive1.Drive  
    DirSave = Dir1.Path  
    SendString = ""  
    Unload FileSelect  
End Sub
```

```
Private Sub Dir1_Change()  
    File1.Path = Dir1.Path  
End Sub
```

```
Private Sub Drive1_Change()  
    Dir1.Path = Drive1.Drive  
End Sub
```

```
Private Sub File1_Click()  
    Text1.Text = Dir1.Path  
    If Right$(Text1.Text, 1) <> "\" Then  
        Text1.Text = Text1.Text & "\"  
    End If  
    Text1.Text = Text1.Text & File1  
End Sub
```

```
Private Sub Form_Load()  
    Text1.Text = "*.mdb"  
    If DriveSave = "" Then  
        Drive1.Drive = "C:"  
        Dir1.Path = "\"  
    Else  
        Drive1.Drive = DriveSave  
        Dir1.Path = DirSave  
    End If  
End Sub
```

```
Private Sub Text1_Change()  
    On Local Error GoTo Cantdoit  
    File1.Pattern = Text1.Text  
Exit Sub
```

```
Cantdoit:  
Exit Sub  
End Sub
```

```
Private Sub Text1_KeyPress(KeyAscii As Integer)  
    On Local Error GoTo Cantdoit  
    If KeyAscii = 13 Then  
        File1.Pattern = Text1.Text  
    End If  
Exit Sub
```

Cantdoit:
Exit Sub
End Sub

FORM1.FRM

VERSION 4.00

Begin VB.Form Form1

 Caption = "Form1"
 ClientHeight = 4230
 ClientLeft = 1095
 ClientTop = 1515
 ClientWidth = 6720
 Height = 4635
 Left = 1035
 LinkTopic = "Form1"
 ScaleHeight = 4230
 ScaleWidth = 6720
 Top = 1170
 Width = 6840

End

Attribute VB_Name = "Form1"

Attribute VB_Creatable = False

Attribute VB_Exposed = False

HELP.FRM

VERSION 4.00

Begin VB.Form Help

```

Caption      = "Help"
ClientHeight = 6060
ClientLeft   = 1500
ClientTop    = 975
ClientWidth  = 6690
ClipControls = 0   'False
ControlBox   = 0   'False
Height       = 6465
Left         = 1440
LinkTopic    = "Form1"
MaxButton    = 0   'False
MinButton    = 0   'False
ScaleHeight  = 6060
ScaleWidth   = 6690
Top          = 630
Width        = 6810

```

Begin VB.CommandButton Command1

```

Caption      = "&Return to Menu"
Default      = -1   'True
Height       = 495
Left         = 2280
TabIndex     = 2
Top          = 5520
Width        = 2055

```

End

Begin VB.Frame Frame1

```

Height       = 5415
Left         = 0
TabIndex     = 0
Top          = 0
Width        = 6615

```

Begin VB.TextBox Text1

```

Height       = 5055
Left         = 120
Locked       = -1   'True
MultiLine    = -1   'True
ScrollBars   = 2   'Vertical
TabIndex     = 1
Text         = "HELP.frx":0000
Top          = 240
Width        = 6375

```

End

End

End

Attribute VB_Name = "Help"

Attribute VB_Creatable = False

Attribute VB_Exposed = False

Option Explicit

Sub LoadScreen()

'This routine will load a screen from the CAST database
'and display it in a text box on this 'Help' form.

```

CastTable.Index = "Primary"
CastTable.Seek "=", SendInt
If Not CastTable.NoMatch Then
    Text1.Text = CastTable("TextInfo")
End If

```

End Sub

Private Sub Command1_Click()
 Unload Help
End Sub

Private Sub Form_Load()
 Call LoadScreen
End Sub

ITEMWINDOW.FRM

VERSION 4.00

Begin VB.Form ItemWindow

```

Appearance      = 0   'Flat
AutoRedraw      = -1  'True
BackColor       = &H00FFFFFF&
ClientHeight    = 4020
ClientLeft      = 1650
ClientTop       = 1455
ClientWidth     = 7365
ClipControls    = 0   'False
ControlBox      = 0   'False

```

BeginProperty Font

```

name           = "MS Sans Serif"
charset        = 1
weight         = 700
size           = 8.25
underline      = 0   'False
italic         = 0   'False
striketrough   = 0   'False

```

EndProperty

```

ForeColor      = &H80000008&
Height         = 4425
KeyPreview     = -1  'True
Left           = 1590
LinkTopic      = "Form1"
MaxButton      = 0   'False
MinButton      = 0   'False
ScaleHeight    = 4020
ScaleWidth     = 7365
Top            = 1110
WhatsThisButton = -1  'True
WhatsThisHelp  = -1  'True
Width          = 7485
WindowState    = 2   'Maximized

```

Begin VB.CommandButton Command4

```

BackColor      = &H00808080&
Caption        = "&Help"
Height         = 495
Left           = 7560
TabIndex       = 6
TabStop        = 0   'False
Top            = 6120
Width          = 1695

```

End

Begin VB.OptionButton Option1

BeginProperty Font

```

name           = "Times New Roman"
charset        = 1
weight         = 700
size           = 14.25
underline      = 0   'False
italic         = 0   'False
striketrough   = 0   'False

```

EndProperty

```

Height         = 330
Index          = 2
Left           = 5280
TabIndex       = 1
TabStop        = 0   'False

```

```

Top           = 3240
Width        = 3495
End

Begin VB.OptionButton Option1
BeginProperty Font
    name           = "Times New Roman"
    charset        = 1
    weight         = 700
    size           = 14.25
    underline      = 0 'False
    italic         = 0 'False
    strikethrough  = 0 'False
EndProperty
Height        = 330
Index         = 5
Left          = 5280
TabIndex      = 4
TabStop       = 0 'False
Top           = 5040
Width         = 3495
End

```

```

Begin VB.OptionButton Option1
BeginProperty Font
    name           = "Times New Roman"
    charset        = 1
    weight         = 700
    size           = 14.25
    underline      = 0 'False
    italic         = 0 'False
    strikethrough  = 0 'False
EndProperty
Height        = 330
Index         = 4
Left          = 5280
TabIndex      = 3
TabStop       = 0 'False
Top           = 4440
Width         = 3495
End

```

```

Begin VB.OptionButton Option1
BeginProperty Font
    name           = "Times New Roman"
    charset        = 1
    weight         = 700
    size           = 14.25
    underline      = 0 'False
    italic         = 0 'False
    strikethrough  = 0 'False
EndProperty
Height        = 330
Index         = 3
Left          = 5280
TabIndex      = 2
TabStop       = 0 'False
Top           = 3840
Width         = 3495
End

```

```

Begin VB.OptionButton Option1

```

```

BeginProperty Font
    name           = "Times New Roman"
    charset        = 1
    weight         = 700
    size           = 14.25
    underline      = 0 'False
    italic         = 0 'False
    strikethrough  = 0 'False
EndProperty
Height           = 330
Index            = 1
Left             = 5280
TabIndex        = 0
TabStop         = 0 'False
Top             = 2640
Width           = 3495
End

Begin VB.CommandButton Command1
    Enabled       = 0 'False
    Height        = 495
    Index         = 5
    Left          = 5160
    TabIndex      = 13
    TabStop       = 0 'False
    Top           = 4920
    Visible       = 0 'False
    Width         = 3735
End

Begin VB.CommandButton Command1
    Enabled       = 0 'False
    Height        = 495
    Index         = 4
    Left          = 5160
    TabIndex      = 12
    TabStop       = 0 'False
    Top           = 4320
    Visible       = 0 'False
    Width         = 3735
End

Begin VB.CommandButton Command1
    Enabled       = 0 'False
    Height        = 495
    Index         = 3
    Left          = 5160
    TabIndex      = 11
    TabStop       = 0 'False
    Top           = 3720
    Visible       = 0 'False
    Width         = 3735
End

Begin VB.CommandButton Command1
    Enabled       = 0 'False
    Height        = 495
    Index         = 1
    Left          = 5160
    TabIndex      = 9
    TabStop       = 0 'False
    Top           = 2520

```

```

Visible      = 0 'False
Width        = 3735
End

Begin VB.CommandButton Command1
Enabled      = 0 'False
Height       = 495
Index        = 2
Left         = 5160
TabIndex     = 10
TabStop      = 0 'False
Top          = 3120
Visible      = 0 'False
Width        = 3735
End

Begin VB.Frame Frame1
Caption      = "Frame1"
Height       = 6615
Left         = 120
TabIndex     = 7
Top          = 120
Width        = 9375
Begin VB.CommandButton Command2
Caption      = "&Next Question"
Height       = 495
Left         = 360
TabIndex     = 5
TabStop      = 0 'False
Top          = 6000
Width        = 2295
End
End
Begin VB.PictureBox ItemPicture
Enabled      = 0 'False
Height       = 5535
Left         = 120
ScaleHeight  = 5475
ScaleWidth   = 8955
TabIndex     = 8
TabStop      = 0 'False
Top          = 240
Width        = 9015
Begin VB.Timer Timer1
Interval     = 1000
Left         = 240
Top          = 1440
End
End
Begin VB.Label Label2
Caption      = "E"
BeginProperty Font
name         = "Times New Roman"
charset      = 1
weight       = 700
size         = 18
underline    = 0 'False
italic       = 0 'False
strikethrough = 0 'False
EndProperty
Height       = 375
Index        = 5
Left         = 4320
TabIndex     = 19

```

```

Top           = 4560
Width        = 255
End
Begin VB.Label Label2
Caption      = "D"
BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 18
    underline  = 0 'False
    italic     = 0 'False
    strikethrough = 0 'False
EndProperty
Height      = 375
Index       = 4
Left        = 4320
TabIndex    = 18
Top         = 3960
Width       = 255
End
Begin VB.Label Label2
Caption      = "C"
BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 18
    underline  = 0 'False
    italic     = 0 'False
    strikethrough = 0 'False
EndProperty
Height      = 375
Index       = 3
Left        = 4320
TabIndex    = 17
Top         = 3360
Width       = 255
End
Begin VB.Label Label2
Caption      = "B"
BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 18
    underline  = 0 'False
    italic     = 0 'False
    strikethrough = 0 'False
EndProperty
Height      = 375
Index       = 2
Left        = 4320
TabIndex    = 16
Top         = 2760
Width       = 255
End
Begin VB.Label Label2
Caption      = "A"
BeginProperty Font
    name      = "Times New Roman"
    charset   = 1

```

```

        weight      = 700
        size        = 18
        underline   = 0   'False
        italic      = 0   'False
        strikethrough = 0   'False
    EndProperty
    Height          = 375
    Index           = 1
    Left            = 4320
    TabIndex        = 15
    Top             = 2160
    Width           = 255
End
Begin VB.Label Label1
    Caption          = "One moment please..."
    BeginProperty Font
        name          = "Times New Roman"
        charset       = 1
        weight        = 700
        size          = 15.75
        underline     = 0   'False
        italic        = 0   'False
        strikethrough = 0   'False
    EndProperty
    Height          = 375
    Left            = 2760
    TabIndex        = 14
    Top             = 1560
    Width           = 3735
End
End
End
Attribute VB_Name = "ItemWindow"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Option Explicit

Const FirstExamineeSlot = 1

Dim ItemBeginTime As Long
Dim ItemEndTime As Long

Dim TestID As Integer
Dim NextTest As Integer
Dim IsSample As Integer
Dim RandomSelection As Integer
Dim TestLength As Integer
Dim ExpPtr As Integer
Dim ExpCount As Integer
Dim ItemNumber As Integer
Dim ItemCount As Integer
Dim Answer As Integer
Dim Response As Integer
Dim ExpItemNumber As Integer
Dim ExperimentalFlag As Integer
Dim OldPointer As Integer
Dim Pointer As Integer
Dim Column As Integer
Dim Correct As Integer
Dim FirstPass As Integer

```

```

Dim Table As String * 1420 '35 x 20 is the largest
Dim Params As String * 3500 '257 x 13 is the largest record

Dim Pointers(1 To MaxItems) As Long
Dim EPointers(1 To MaxItems) As Long
Dim Used(1 To MaxItems) As Integer
Dim ExpUsed(1 To MaxItems) As Integer

Dim A, B, C, PVar, Theta, LL, HH, II, AC, BC As Double

Dim ItemBuffer As String
Dim Rand As Integer
Dim GivingExtra As Integer
Dim NumberOfDistractors As Integer
Dim Processing As Integer
'Dim CurrentARTime As Integer
Dim Seconds As Integer

Sub AdminItem(GetNext As Integer)
'This routine checks the examinee's response to the previous question.
'This routine will also update the system's adaptive predictions, and
'the best new item will be determined.
    Dim X As Integer

    Call StopTimer
    Call GetParams
    Response = Pointer
    Option1(Pointer).Value = False
    If Answer = Response Then
        Correct = True
    Else
        Correct = False
    End If
    If Not ExperimentalFlag Then
        Call UpdateMath
    End If
    If WriteDisk Then
        Call WriteData
    End If
    If EnableShowStats Then
        Call DisplayStats
    End If
    If GetNext Then
        Call UpdateStrategy
        Call GetColumn
        Call GetItemNumber
    End If
    For X = 1 To MaxDistractors
        Label2(X).Visible = False
    Next X
    Call SystemPause(0.5)
    Call StartTimer
    DoEvents
End Sub

```

```

Sub CheckRestart()
'This routine will initialize the 'Items Used' array, and then if this
'test administration has been restarted, then the current
'status of the administration will be determined in order to
'properly resume.
    Dim X, EndFound As Integer
    Dim Temp As String
    Dim Result As String

    Call InitUsed
    If Restart Then
        ItemCount = 1
        ExpPtr = 1
        ExpCount = 1
        Temp = OutputString
        X = InStr(1, OutputString, "ARITHMETIC REASONING:", 1)
        If X <> 0 Then 'Aritmetic Reasoning
            NextTest = AR
            EndFound = False
            While Not EndFound ' Cycle past ARITHMETIC REASONING Label
                EndFound = ParseString(Temp, Result)
                If Mid$(Result, 4, 1) = " " And Mid$(Result, 6, 1) = " " Then
                    If Mid$(Result, 5, 1) = "N" Then 'Not experimental
                        Theta = Val(Mid$(Result, 13, 10))
                        PVar = Val(Mid$(Result, 24, 9))
                    End If
                ElseIf InStr(1, Result, "ARITHMETIC REASONING:", 1) > 0 Then
                    EndFound = True
                    WKTheta = Theta
                    WKPvar = PVar
                End If
            Wend
        Else
            NextTest = WK
        End If
        PVar = InitPVAR(NextTest)
        Theta = InitTheta(NextTest)
        EndFound = False
        While Not EndFound
            EndFound = ParseString(Temp, Result)
            If Not EndFound Then
                If Mid$(Result, 4, 1) = " " And Mid$(Result, 6, 1) = " " Then
                    If Mid$(Result, 5, 1) = "N" Then 'Not experimental
                        X = Val(Left$(Result, 3))
                        If X > 0 Then
                            Used(X) = True
                        End If
                        ItemCount = ItemCount + 1
                        Theta = Val(Mid$(Result, 13, 10))
                        PVar = Val(Mid$(Result, 24, 9))
                        Seconds = Seconds + Val(Mid$(Result, 33, 4))
                    ElseIf Mid$(Result, 5, 1) = "Y" Then 'Experimental
                        If WKExp(ExpPtr, 2) = ItemCount Then
                            ExpCount = ExpCount + 1
                            If ExpCount = WKExp(ExpPtr, 1) Then
                                ExpPtr = ExpPtr + 1
                                ExpCount = 1
                            End If
                        End If
                        X = Val(Left$(Result, 3))
                    End If
                End If
            End If
        End While
    End Sub

```

```

                If X > 0 Then
                    ExpUsed(X) = True
                End If
            End If
        End If
    End If
Wend
If NextTest = WK And ItemCount >= WKLength Then
    NextTest = AR
    ItemCount = 1
End If
End If
End Sub

```

```

Sub ClearOptions()
'This routine simply clears the 'option' tool. This is done to eliminate
'any item from being 'selected'.
    Dim X As Integer

```

```

    For X = 1 To MaxDistractors
        Option1(X).Value = False
        Option1(X).ForeColor = BLACK
    Next X
    OldPointer = 0
    If Command2.Visible Then
        Command2.SetFocus
    End If
End Sub

```

```

Private Sub DisplayStats()
'DisplayStats will display the critical values relating to the\
'current test administration on the screen. DisplayStats is only
'used in testing procedures.
    Dim Z As Integer

```

```

    ItemPicture.Cls
    ItemPicture.Print "Item Count "; ItemCount
    ItemPicture.Print "InfoTable Column:"; Column
    ItemPicture.Print "Random pick from best "; RandomSelection
    For Z = 1 To RandomSelection
        ItemPicture.Print RNDItems(Z)
    Next Z
    ItemPicture.Print "Random Seed "; Rand
    If TestID = WK Then
        ItemPicture.Print "WK ";
    Else
        ItemPicture.Print "AR ";
    End If
    If ExperimentalFlag Then
        ItemPicture.Print "EXP Item Number "; ExpItemNumber
    End If
    ItemPicture.Print "Item "; ItemNumber
    ItemPicture.Print "A= "; A;
    ItemPicture.Print ", B= "; B;
    ItemPicture.Print ", C= "; C
    ItemPicture.Print "RESPONSE "; Response
    ItemPicture.Print "ANSWER KEY "; Answer
    If Correct Then
        ItemPicture.Print "Correct answer"
    Else
        ItemPicture.Print "Wrong answer"
    End If

```

```

End If
ItemPicture.Print "Theta      "; Theta
ItemPicture.Print "PV        "; PVar
MsgBox ""
End Sub

```

```

Sub EvaluateKey(KeyCode As Integer)
'This routine will check the key the user entered and determine if
'the user intended to select a distractor.

```

```

    If Processing Then
        Exit Sub
    End If

```

```

    On Local Error GoTo SetToTrue
    If KeyCode = 9 Then
        KeyCode = OldPointer + 1
        If KeyCode > MaxDistractors Then
            KeyCode = 1
        End If
        Call Option1_Click(KeyCode)
        Exit Sub
    End If

```

```

    If KeyCode >= 97 Then
        KeyCode = KeyCode - 32
    End If

```

```

    If KeyCode = 13 Then
        Call Command2_Click
        Exit Sub
    End If

```

```

    KeyCode = KeyCode - 64
    If KeyCode >= 1 And KeyCode <= NumberOfDistractors Then
        Call Option1_Click(KeyCode)
        DoEvents
        Option1(KeyCode).Value = True
        DoEvents
        Option1(KeyCode).SetFocus
        DoEvents
        Exit Sub
    End If
    Exit Sub

```

```

SetToTrue:
    Option1(KeyCode).Value = True
    DoEvents
    Resume
End Sub

```

```

Sub FinalCalculations(Timeout As Integer)
'This routine performs the final calculations necessary to determine
'the predicted AFQT (and other) scores.

```

```

    Dim N8CastEAFQT As Double
    Dim N8CastWK As Double
    Dim N8CastAR As Double
    Dim N8CastPWK As Double
    Dim N8CastPAR As Double

```

```

    P1 = 1 / (1 + Exp(ProbParams(1) * WKTheta - ProbParams(2) * ARTheta +
    ProbParams(3)))
    If P1 > 0.99 Then

```

```

        P1 = 0.99
        P2 = 0.01
    Else
        P2 = (1 / (1 + Exp(ProbParams(4) * WKTheta - ProbParams(5) * ARTheta -
ProbParams(6)))) - P1
    End If
    P1 = Int(P1 * 100 + 0.5)
    P2 = Int(P2 * 100 + 0.5)
    P3 = 100 - (P1 + P2)

' **** DEVELOP AFQT SCORE ****
N8CastEAFQT = RegCoefWK * WKTheta + RegCoefAR * ARTheta + RegConst
' **** DEVELOP FINAL SCORES AS TWO-CHARACTER STRINGS ****
AFQT = N8CastEAFQT
If AFQT < 1 Then
    AFQT = 1
End If
' Correct WkTheta and ARTheta to 1980 Youth Population
' using mean and sd data from Wise et al. 1989 report on CAST.
AWK = 50 + (((WKTheta + 0.247) / 0.602) * 10)
AAR = 50 + (((ARTheta + 0.213) / 0.732) * 10)
If AWK < 20 Then
    AWK = 20
End If
If AAR < 26 Then
    AAR = 26
End If
If AWK > 61 Then
    AWK = 61
End If
If AAR > 66 Then
    AAR = 66
End If
End Sub

```

```

Function GetCurrentAFQT() As Double
'The current AFQT score is determined using the following formula:
    GetCurrentAFQT = RegCoefWK * WKTheta + RegCoefAR * Theta + RegConst
End Function

```

```

Sub GetTimeoutAnswer()
'Quickly get the answer to an item. Needed when a timeout occurs
'and the correct examinee record must be constructed (containing
'the correct answer for each item). Timeouts are handled by
'completing the test getting the rest of the answers wrong.
    Dim Finished As Integer
    Dim Ptr As Integer
    Dim Char As String * 1

    Call LoadItem(ItemNumber, TestID)
    Finished = False
    Ptr = 1
    While Not Finished
        Char = Mid$(ItemBuffer, Ptr, 1)
        Ptr = Ptr + 1
        If Char > Chr$(0) And Char < Chr$(9) Then
            Answer = Asc(Char)
            Finished = True
        End If
    Wend

```

End Sub

Function PercentileFunction(Z As Double) As Double

'This function returns a percentile.

Dim FL As Double

Dim Q As Double, P As Double

FL = 0: Rem (PERCENTILE FUNCTION)

If Z < 0 Then FL = 1: Z = Abs(Z)

Q = 1! + 0.196854 * Z + 0.115194 * Z * Z + 0.000344 * Z ^ 3 + 0.019527 * Z

^ 4

Q = Q ^ 4

P = 1 - 1 / (2 * Q)

If FL = 1 Then P = 1 - P

If P < 0.01 Then P = 0.01

PercentileFunction = P

End Function

Sub FinalRoutines(Timeout As Integer)

'This routine calls most of the final routines that determine:

'the final test administration outcome, the final data to be written to the

'file and then the test feedback procedures are called.

Dim X As Integer

TestID = FINAL

ItemPicture.Cls

For X = 1 To MaxDistractors

Command1(X).Visible = False

Option1(X).Visible = False

Label2(X).Visible = False

Next X

Label1.Visible = True

DoEvents

ARTheta = Theta

ARPvar = PVar

Call FinalCalculations(Timeout)

Call GetPerformance(OutputString)

OutputString = OutputString & Date\$ & " " & Time\$ & CRLF

OutputString = OutputString & "Complete:" & CRLF

OutputString = OutputString & Format\$(AFQT, "###") & "," & Format\$(AWK, "###") & "," & Format\$(AAR, "###") & "," & Str\$(P1) & "," & Str\$(P2) & "," & Str\$(P3)

If WriteDisk Then

Call WriteBuffer(True)

End If

SendString = "1"

Feedback.Show MODAL

SendString = "2"

Feedback.Show MODAL

End Sub

Sub GetExpItemNumber()

'Get an experimental item number

Dim Temp, Rand As Integer

If TestID = WK Then

Temp = MaxWKExp

Else

Temp = MaxARExp

End If

Rand = 0

```

While Rand < 1 Or Rand > Temp
    Rand = Int(Rnd * 100)
    If Rand > 0 Then
        If Not ExpUsed(Rand) And Rand > 0 And Rand <= Temp Then
            ExpUsed(Rand) = True
            ExpItemNumber = Rand
        Else
            Rand = 0
        End If
    End If
Wend
ExperimentalFlag = True
End Sub

Sub IntroCorrect()
    'This routine simply displays a message on the screen that tells the
    'examinee that the sample question was answered correctly.
    Banner = "Good, " & First & ", "
    BannerX = 1700
    BannerY = 900
    If NextTest = WK Then
        SendString = Str$(INTRO) & ",11,11"
    Else
        SendString = Str$(INTRO) & ",9,9"
    End If
    SendInt = ALLOW_BACKUP
    Dialog.Show MODAL
    Banner = ""
End Sub

Sub IntroInCorrect()
    'This routine simply displays a message on the screen that tells the
    'examinee that the sample question was answered incorrectly.
    If NextTest = WK Then
        SendString = Str$(INTRO) & ",12,12"
    Else
        SendString = Str$(INTRO) & ",10,10"
    End If
    SendInt = 0
    Dialog.Show MODAL
End Sub

Sub ReDisplayScreen()
    'This routine will render several buttons unavailable for re-display
    'purposes.
    Dim X
    For X = 1 To MaxDistractors
        Command1(X).Visible = False
        Option1(X).Visible = False
        Option1(X).Caption = ""
        Option1(X).Value = False
        Option1(X).ForeColor = BLACK
        Label2(X).Visible = False
    Next X
    DoEvents
    Call ShowItem
End Sub

```

```

Sub WriteBuffer(CompleteFlag As Integer)
'The current test results supplied by the examinee are written to the
'EXAMINEE database.
    ExamineeTable.Edit
    If CompleteFlag Then
        ExamineeTable("CompleteFlag") = True
    End If
    ExamineeTable("TestInfo") = OutputString
    ExamineeTable.Update
End Sub

Sub WriteTimeoutRecord()
'If a timeout condition is detected, the system will act like the examinee
'completed the test, however, got all the remaining questions wrong.
    Dim CorrectMark As String * 1
    Dim Temp As String
    Dim ItemLoop As Integer

    GoSub TimeoutProcess
    ExperimentalFlag = False
    If WriteDisk Then
        For ItemLoop = ItemCount + 1 To TestLength
            Call UpdateStrategy
            Call GetColumn
            Call GetItemNumber
            Call GetTimeoutAnswer
            Call StartTimer
            GoSub TimeoutProcess
        Next ItemLoop
    End If
    Exit Sub

TimeoutProcess:
    Call StopTimer
    Call GetParams
    Response = 0
    Correct = False
    If Not ExperimentalFlag Then
        Call UpdateMath
    End If
    If WriteDisk Then
        Call WriteData
    End If
    Return
End Sub

Private Sub Command1_Click(Index As Integer)
    Option1(Index).Value = True
    Call Option1_Click(Index)
End Sub

Private Sub Command1_KeyPress(Index As Integer, KeyAscii As Integer)
    If Processing Then
        KeyAscii = 0
    Else
        Call EvaluateKey(KeyAscii)
    End If
End Sub

Private Sub Command2_Click()
    Dim EndedWithExp As Integer

```

```

Dim X As Integer
Dim ProvisionalAFQT As Double

Processing = True
For X = 1 To NumberOfDistractors
    If Option1(X).Value Then
        Pointer = X
        X = NumberOfDistractors
    End If
Next X

If Pointer = 0 Then
    MsgBox "Please select an answer before you press the 'Next Question'
button."
    Processing = False
    DoEvents
    Call ClearOptions
    Exit Sub
End If

EndedWithExp = False
If IsSample Then
    ItemPicture.Cls
    If TestID = INTRO Then
        If Pointer = Answer Then
            Call IntroCorrect
            If SendInt = ALLOW_BACKUP Then
                SendInt = 0
                X = OldPointer
                Call ShowItem
                OldPointer = X
                Processing = False
                Exit Sub
            Else
                TestID = NextTest
                Call SubtestInit
                IsSample = False
            End If
        End If
        Processing = False
        Exit Sub
    Else
        Call IntroInCorrect
        X = OldPointer
        Call ShowItem
        OldPointer = X
        Processing = False
        Exit Sub
    End If
End If

If AllItemDebug Then
    Set ExperimentalFlag if I want EXPs
    ExperimentalFlag = True

    ItemNumber = ItemNumber + 1
    If ExperimentalFlag Then
        If TestID = WK Then
            TestLength = MaxWKExp
        Else

```

```

        TestLength = MaxARExp
    End If
    ExpItemNumber = ItemNumber
End If
If ItemNumber > TestLength Then
    If TestID = WK Then
        TestID = AR
        Call SubtestInit
        Processing = False
        Exit Sub
    Else
        Unload ItemWindow
        Processing = False
        Exit Sub
    End If
End If
If ExperimentalFlag Then
    Call LoadItem(ExpItemNumber, TestID + 2)
End If
Else
    If ExperimentalFlag Then
        ExpCount = ExpCount + 1
        If TestID = WK Then
            If ExpCount > WKExp(ExpPtr, 1) Then
                Call AdminItem(False)
                ExperimentalFlag = False
                ExpPtr = ExpPtr + 1
                ExpCount = 1
                If ItemCount + 1 <= TestLength Then
                    Call LoadItem(ItemNumber, TestID)
                    Call ShowItem
                    Processing = False
                    Exit Sub
                Else
                    EndedWithExp = True
                End If
            Else
                Call AdminItem(False)
                Call GetExpItemNumber
                Call LoadItem(ExpItemNumber, TestID + 2)
                Call ShowItem
                Processing = False
                Exit Sub
            End If
        Else
            If ExpCount > ARExp(ExpPtr, 1) Then
                Call AdminItem(False)
                ExperimentalFlag = False
                ExpPtr = ExpPtr + 1
                ExpCount = 1
                If ItemCount + 1 <= TestLength Then
                    Call LoadItem(ItemNumber, TestID)
                    Call ShowItem
                    Processing = False
                    Exit Sub
                Else
                    EndedWithExp = True
                End If
            Else
                Call AdminItem(False)
                Call GetExpItemNumber
                Call LoadItem(ExpItemNumber, TestID + 2)
            End If
        End If
    End If

```

```

        Call ShowItem
        Processing = False
        Exit Sub
    End If
End If
End If
If ItemCount + 1 > TestLength Then
    If TestID = WK Then
        If Not GivingExtra And Theta < ExtraValue(1) And ExtraItems(1) > 0
Then
            GivingExtra = True
            TestLength = TestLength + ExtraItems(1)
            Call AdminItem(True)
        Else
            If WKExpAdmin > 0 Then
                If WKExp(ExpPtr, 2) = ItemCount + 1 Then
                    Call AdminItem(False)
                    Call GetExpItemNumber
                    Call LoadItem(ExpItemNumber, TestID + 2)
                    Call ShowItem
                    Processing = False
                    Exit Sub
                End If
            End If
            If Not EndedWithExp Then
                Call AdminItem(False)
            End If
            If WriteDisk Then
                OutputString = OutputString & "ARITHMETIC REASONING:" & CRLF
                Call WriteBuffer(False)
            End If
            WKTheta = Theta
            WKPvar = PVar
            ItemPicture.Cls
            OutputString = OutputString & "Instruction Sequence Begin:"
            OutputString = OutputString & Date$ & " " & Time$ & CRLF
            If WriteDisk Then
                Call WriteBuffer(False)
            End If
            SendString = Str$(INTRO) & ",14,14"
            SendInt = 0
            Dialog.Show MODAL
            SendString = Str$(INTRO) & ",15,16"
            SendInt = 0
            Dialog.Show MODAL
            IsSample = True
            TestID = INTRO
            NextTest = AR
            Call LoadItem(102, TestID)
            Call ShowItem
            OutputString = OutputString & "Instruction Sequence End: "
            OutputString = OutputString & Date$ & " " & Time$ & CRLF
            If WriteDisk Then
                Call WriteBuffer(False)
            End If
            Processing = False
            Exit Sub
        End If
    Else 'AR
        ProvisionalAFQT = GetCurrentAFQT()
        If (ProvisionalAFQT > ExtraValue(2)) And (ProvisionalAFQT <
ExtraValue(3)) And (TestLength < ARLength + ExtraItems(2)) Then

```

```

0 Then      'If Not GivingExtra And Theta < ExtraValue(2) And ExtraItems(2) >
            GivingExtra = True
            TestLength = TestLength + 1
            Call AdminItem(True)
        Else
            If ARExpAdmin > 0 Then
                If ARExp(ExpPtr, 2) = ItemCount + 1 Then
                    Call AdminItem(False)
                    Call GetExpItemNumber
                    Call LoadItem(ExpItemNumber, TestID + 2)
                    Call ShowItem
                    Processing = False
                    Exit Sub
                End If
            End If
            If Not EndedWithExp Then
                Call AdminItem(False)
            End If
            Call FinalRoutines(False)
            Unload ItemWindow
            Processing = False
            Exit Sub
        End If
    End If
Else
    Call AdminItem(True)
    If TestID = AR Then
        'If CurrentARTime >= ARTimeLimit Then
        '    Call FinalRoutines(False)
        '    Unload ItemWindow
        '    Exit Sub
        'End If
    End If
End If
If Not ExperimentalFlag Then
    ItemCount = ItemCount + 1
End If
If TestID = WK And WKExpAdmin > 0 Then
    If WKExp(ExpPtr, 2) = ItemCount Then
        Call GetExpItemNumber
    End If
ElseIf TestID = AR And ARExpAdmin > 0 Then
    If ARExp(ExpPtr, 2) = ItemCount Then
        Call GetExpItemNumber
    End If
End If
If ExperimentalFlag Then
    Call LoadItem(ExpItemNumber, TestID + 2)
End If
End If
If Not ExperimentalFlag Then
    Call LoadItem(ItemNumber, TestID)
End If
Call ShowItem
Processing = False
End Sub

Private Sub Command2_KeyPress(KeyAscii As Integer)
    If Processing Then
        KeyAscii = 0
    Else

```

```

        Call EvaluateKey(KeyAscii)
    End If
End Sub

```

```

Private Sub Command4_Click()
    Dim X As Integer
    Dim Switch As Integer

    Switch = 0
    For X = 1 To NumberOfDistractors
        If Option1(X).Value Then
            Switch = X
            X = NumberOfDistractors
        End If
    Next X

    ItemEndTime = (Timer - ItemBeginTime)
    SendInt = HELPSCREEN
    SendString = Str$(INTRO) & ",99,99"
    Dialog.Show MODAL
    If SendInt = QUIT Then
        If WriteDisk Then
            CRLF
            OutputString = OutputString & "Interrupt:" & Date$ & " " & Time$ &
            Call WriteBuffer(False)
        End If
        Unload ItemWindow
    End If
    Call ReDisplayScreen
    If Switch > 0 Then
        Call Option1_Click(Switch)
    End If
    ItemBeginTime = Timer - ItemEndTime
End Sub

```

```

Private Sub Command4_GotFocus()
    Dim X As Integer

    For X = 1 To MaxDistractors
        Option1(X).Value = False
        Option1(X).ForeColor = BLACK
    Next X
    OldPointer = 0
    Command4.Caption = "&HELP"
End Sub

```

```

Private Sub Command4_KeyPress(KeyAscii As Integer)
    If Processing Then
        KeyAscii = 0
    Else
        Call EvaluateKey(KeyAscii)
    End If
End Sub

```

```

Private Sub Command4_LostFocus()
    Command4.Caption = "&Help?"
End Sub

Private Sub Form_Activate()
    If Not DebugFlag Then
        On Error GoTo GeneralError
    End If
    If FirstPass Then
        FirstPass = False
        ItemWindow.ForeColor = QBColor(FColor)
        ItemWindow.BackColor = QBColor(BColor)
        Call CheckRestart
        Call TestIntro
    End If
Exit Sub

GeneralError:
    MsgBox "An error has occurred."
    Unload ItemPicture
End Sub

Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
    If Processing Then
        KeyCode = 0
    End If
End Sub

Private Sub Form_KeyPress(KeyAscii As Integer)
    If Processing Then
        If KeyAscii = 13 Then
            MsgBox "Please do not hold the ENTER key down."
            KeyAscii = 0
            Call SystemPause(0.5)
        End If
    End If
    'Call EvaluateKey(KeyAscii)
End Sub

Private Sub Form_KeyUp(KeyCode As Integer, Shift As Integer)
    If Processing Then
        KeyCode = 0
    End If
End Sub

Private Sub Form_Load()
    Seconds = 0
    TestID = INTRO
    NextTest = WK
    Banner = ""
    ItemPicture.FontSize = ItemFontSize
    FirstPass = True
    Label1.Visible = False
    Frame1.Caption = ""
    SendInt = 0
    Processing = False
End Sub

```

```

Private Sub GetColumn()
'This routine will calculate the correct column in the information table
'based on the examinee's current performance. Bounds checking are also
'performed.
    Column = Int(AC * Theta + BC + 0.5)
    If Column < 1 Then Column = 1
    If Column > II Then Column = II
End Sub

Private Sub GetItemNumber()
'Get an item number
    Dim Ptr, GotOne As Integer
    Dim HiByte, Z As Integer

    Rand = 0
    While Rand < 1 Or Rand > RandomSelection
        Rand = Int(Rnd * 10)
    Wend
    GotOne = False
    While Not GotOne
        GoSub GetTheItemNumber
        If Not GotOne Then
            Column = Column - 1
            If Column < 1 Then
                MsgBox "Info table error: Looking for row 0"
            End If
        End If
    Wend
    Used(ItemNumber) = True
Exit Sub

GetTheItemNumber:
    Ptr = ((InfoRows * (Column - 1)) * 2) + 1
    For Z = 1 To RandomSelection
        GotOne = False
        If Ptr > (InfoRows * InfoColumns * 2) Then
            Return
        End If
        While Not GotOne
            ItemNumber = Asc(Mid$(Table, Ptr, 1))
            HiByte = Asc(Mid$(Table, Ptr + 1, 1))
            While HiByte > 0
                ItemNumber = ItemNumber + 256
                HiByte = HiByte - 1
            Wend
            Ptr = Ptr + 2
            If Not Used(ItemNumber) Then
                RNDItems(Z) = ItemNumber
                GotOne = True
            End If
        Wend
    Next Z
    ItemNumber = RNDItems(Rand)
    ExperimentalFlag = False
    Return
End Sub

Private Sub GetParams()
'Pull the correct parameters out of the buffer named 'Params'.
    Dim Ptr, Z As Integer
    Dim AA$, BB$, CC$

```

```

Ptr = (13 * (ItemNumber - 1)) + 1
AA$ = "": BB$ = "": CC$ = ""
For Z = 1 To 4
    AA$ = AA$ + Mid$(Params, Ptr, 1)
    Ptr = Ptr + 1
Next Z
For Z = 1 To 5
    BB$ = BB$ + Mid$(Params, Ptr, 1)
    Ptr = Ptr + 1
Next Z
For Z = 1 To 4
    CC$ = CC$ + Mid$(Params, Ptr, 1)
    Ptr = Ptr + 1
Next Z
A = Val(AA$): B = Val(BB$): C = Val(CC$)
End Sub

```

```

Private Sub InitUsed()
'Set the entire array holding the items that have been previously used to
'FALSE or unused.
    Dim Z As Integer
    For Z = 1 To MaxItems
        Used(Z) = False
        ExpUsed(Z) = False
    Next Z
End Sub

```

```

Private Sub LoadItem(RecordNumber As Integer, Offset)
'An item from the CAST database is loaded into an item buffer.
    Dim Finished, X, Y, Z, DisFound As Integer
    Dim Distractor As Integer
    Dim EndFound As Integer
    Dim Char As String * 1
    Dim Temp, Junk As String

    Finished = False
    DisFound = False
    ItemBuffer = ""
    Junk = ""
    For X = 1 To MaxDistractors
        Command1(X).Visible = False
        Option1(X).Visible = False
        Option1(X).Caption = ""
        Option1(X).Value = False
        Option1(X).ForeColor = BLACK
        Label2(X).Visible = False
    Next X
    DoEvents

    CastTable.Seek "=", RecordNumber + (Offset * 1000)
    If Not CastTable.NoMatch Then
        Temp = CastTable("TextInfo")
        X = Len(Temp)
        Distractor = 0
        EndFound = False
        For Z = 1 To X
            Char = Mid$(Temp, Z, 1)
            Select Case Char
                Case Chr$(13):
                    If DisFound Then

```

```

        If TestID = AR Then
            Distractor = Distractor + 1
            If Distractor = MaxDistractors + 2 Then
                EndFound = True
            End If
        Else
            If Len(LTrim(RTrim(Junk))) = 1 And Val(Junk) > 0 And
Val(Junk) < 9 Then
                EndFound = True
            End If
        End If
    If EndFound Then
        For Y = Len(ItemBuffer) To 1 Step -1
            If Mid$(ItemBuffer, Y, 1) = Chr$(0) Then
                ItemBuffer = Left$(ItemBuffer, Y - 1)
                ItemBuffer = ItemBuffer & Chr$(Val(Junk))
                Y = 0
            End If
        Next Y
        Z = X + 1
    Else
        ItemBuffer = ItemBuffer & Chr$(0)
        Junk = ""
    End If
Else
    ItemBuffer = ItemBuffer & Chr$(13)
End If
Case Chr$(10):
Case "!":
    DisFound = True
Case "|":
    DisFound = True
Case Else
    ItemBuffer = ItemBuffer & Char
    If DisFound Then
        Junk = Junk & Char
    End If
End Select
Next Z
End If
End Sub

```

```

Private Sub LoadTables(TEST As Integer)
    'The correct info table buffer and parameter buffer are loaded.
    Dim FileNum As Integer
    Dim X As Integer

    FileNum = FreeFile
    If Not DebugFlag Then
        On Local Error GoTo LError
    End If
    If TEST = WK Then
        CastTable.Seek "=", 9002
        Table = CastTable("TextInfo")
        CastTable.Seek "=", 9000
        Params = CastTable("TextInfo")
    Else
        CastTable.Seek "=", 9003
        Table = CastTable("TextInfo")
        CastTable.Seek "=", 9001
        Params = CastTable("TextInfo")
    End If

```

```

Exit Sub

LTErrror:
    MsgBox "Cannot load tables and/or parameters"
    Unload ItemWindow
End Sub

Private Sub MathInit()
'Initialize certain variables used to calculate administer an
'adaptive test to beginning values.
    A = 0
    B = 0
    C = 0
    LL = -2.25
    HH = 2.25
    II = InfoColumns
    AC = (II - 1) / (HH - LL)
    BC = (HH - II * LL) / (HH - LL)
End Sub

Private Sub ShowItem()
'Show the item that is currently being held in the buffer.
    Dim Z, HTab, VTab, Ptr As Integer
    Dim Finished, Distractor As Integer
    Dim Char As String * 1
    Dim Temp As String
    Dim VTabStart, HTabStart As Integer

    If Not DebugFlag Then
        On Local Error GoTo ShowItemError
    End If

    If TestID = AR Then
        VTabStart = 10 * Screen.TwipsPerPixelY
        HTabStart = Screen.TwipsPerPixelX
    Else
        VTabStart = 30 * Screen.TwipsPerPixelY
        HTabStart = 5 * Screen.TwipsPerPixelX
    End If
    ItemPicture.Cls
    Finished = False
    Distractor = 0
    HTab = HTabStart: VTab = VTabStart
    ItemPicture.CurrentX = HTab * Screen.TwipsPerPixelX
    ItemPicture.CurrentY = VTab
    Ptr = 1
    While Not Finished
        Char = Mid$(ItemBuffer, Ptr, 1)
        Ptr = Ptr + 1
        If Char = Chr$(0) Then
            Distractor = Distractor + 1
            Label2(Distractor).Visible = True
            Option1(Distractor).Caption = " "
        ElseIf Char > Chr$(0) And Char < Chr$(9) Then
            Answer = Asc(Char)
            Finished = True
        ElseIf Char = Chr$(13) Then
            HTab = HTabStart
            VTab = VTab + TextHeight("A") + 70
            ItemPicture.CurrentX = HTab * Screen.TwipsPerPixelX

```

```

        ItemPicture.CurrentY = VTab
    Else
        If Char = Chr$(InverseOn) Then
            ItemPicture.FontUnderline = True
            ItemPicture.ForeColor = QBColor(DColor)
            Temp = ""
            While Char <> Chr$(InverseOff)
                Char = Mid$(ItemBuffer, Ptr, 1)
                Ptr = Ptr + 1
                If Char <> Chr$(InverseOff) Then
                    ItemPicture.Print Char;
                End If
            Wend
            ItemPicture.FontUnderline = False
            ItemPicture.ForeColor = FColor
        Else
            If Distractor > 0 Then
                Option1(Distractor).Caption = Option1(Distractor).Caption &
Char
            Else
                ItemPicture.Print Char;
            End If
        End If
    End If
Wend
OldPointer = 0
Pointer = 0
If TestID = INTRO Then
    Frame1.Caption = "Sample"
Else
    If AllItemDebug Then
        Frame1.Caption = "Item Number: " & Str$(ItemNumber)
    Else
        Frame1.Caption = "Item Number: " & Str$(ItemCount)
        If ExperimentalFlag Then
            Frame1.Caption = Frame1.Caption & " Exp."
        End If
    End If
End If
NumberOfDistractors = Distractor
For Z = 1 To NumberOfDistractors
    Command1(Z).Visible = True
    Option1(Z).Visible = True
Next Z
DoEvents
Exit Sub

ShowItemError:
    MsgBox "An error was encountered while trying to display item #: " &
Str$(ItemNumber)
End Sub

Private Sub StartTimer()
    'Capture beginning 'Timer' value
    ItemBeginTime = Timer
End Sub

Private Sub StopTimer()
    'Capture ending 'Timer' value
    ItemEndTime = (Timer - ItemBeginTime)
End Sub

```

```

Private Sub SubtestInit()
'Initialize the subtest (WK or AR) with starting values and
'proper indentification inside the EXAMINEE record. Also, the
'first item is selected, loaded, and displayed.
    RandomSelection = 5
    GivingExtra = False

    If TestID = WK Then
        If AllItemDebug Then
            TestLength = MaxWKItems
        Else
            TestLength = WKLength
            If WriteDisk Then
                OutputString = OutputString & "WORD KNOWLEDGE:" & CRLF
                Call WriteBuffer(False)
            End If
        End If
    Else
        If AllItemDebug Then
            TestLength = MaxARItems
        Else
            TestLength = ARLength
        End If
    End If
    OutputString = OutputString & Date$ & " " & Time$ & CRLF
    If WriteDisk Then
        Call WriteBuffer(False)
    End If
    If Not Restart Then
        ExpPtr = 1
        ExpCount = 1
        PVar = InitPVAR(TestID)
        Theta = InitTheta(TestID)
    End If
    Call LoadTables(TestID)
    Call MathInit

    ExperimentalFlag = False
    If AllItemDebug Then
        ItemNumber = 1
    Else
        Call GetColumn
        Call GetItemNumber
        If Not Restart Then
            ItemCount = 1
        End If
    End If

    If Restart Then
        Restart = False
    End If
    If TestID = WK And WKExpAdmin > 0 Then
        If WKExp(ExpPtr, 1) > 0 And WKExp(ExpPtr, 2) = ItemCount Then
            Call GetExpItemNumber
        End If
    End If
    If TestID = AR And ARExpAdmin > 0 Then
        If ARExp(ExpPtr, 1) > 0 And ARExp(ExpPtr, 2) = ItemCount Then
            Call GetExpItemNumber
        End If
    End If
    If ExperimentalFlag Then

```

```

    Call LoadItem(ExpItemNumber, TestID + 2)
Else
    Call LoadItem(ItemNumber, TestID)
End If
Call ShowItem
Call StartTimer
If TestID = AR Then
    Seconds = 0
End If
End Sub

```

```

Private Sub TestIntro()

```

```

'This routine will call a series of screens located in the CAST database
'file that are used to present an introduction sequence to the examinee.

```

```

    Dim Z As Integer
    Const IntroScreens = 10

    If Not DebugFlag Then
        On Local Error GoTo TestIntroError
    End If
    OutputString = OutputString & "Instruction Sequence Begin:"
    OutputString = OutputString & Date$ & " " & Time$ & CRLF
    If WriteDisk Then
        Call WriteBuffer(False)
    End If
    IsSample = True

    If Restart And NextTest = AR Then
        SendString = Str$(INTRO) & "," & Str$(IntroScreenStart) & "," &
Str$(IntroScreenEnd - 2)
    Else
        SendString = Str$(INTRO) & "," & Str$(IntroScreenStart) & "," &
Str$(IntroScreenEnd)
    End If
    SendInt = 0
    Dialog.Show MODAL
    If Restart And NextTest = AR Then
        SendString = Str$(INTRO) & ",15,16"
        SendInt = 0
        Dialog.Show MODAL
    End If
    If Restart And NextTest = AR Then
        Call LoadItem(102, TestID)
    Else
        Call LoadItem(101, TestID)
    End If
    Call ShowItem
    OutputString = OutputString & "Instruction Sequence End: "
    OutputString = OutputString & Date$ & " " & Time$ & CRLF
    If WriteDisk Then
        Call WriteBuffer(False)
    End If
    Exit Sub

```

```

TestIntroError:

```

```

    MsgBox "Cannot find: " & IntroImage
    Unload ItemWindow
End Sub

```

```

Private Sub UpdateMath()

```

```

'All adaptive test variables are appropriately updated.

```

```

Dim SS, S, XX, YY, ZZ, Temp, DD, PP As Double

SS = PVar + 1 / (A * A)
S = Sqr(SS)
XX = (Theta - B) / S
ZZ = Abs(XX)
Temp = 1 / (1 + 0.2316419 * ZZ)
YY = -XX * XX / 2
If YY > 85 Then YY = 85
If YY < -85 Then YY = -85
DD = 0.3989423 * Exp(YY)
PP = 1! - DD * Temp * (((1.330274 * Temp - 1.821256) * Temp + 1.781478) *
Temp - 0.3565638) * Temp + 0.3193815)
If XX < 0 Then PP = 1 - PP
If Correct Then
    ZZ = DD / (PP + C / (1 - C))
Else
    ZZ = -DD / (1 - PP)
End If
Theta = Theta + ZZ * PVar / S
PVar = PVar - PVar * PVar * ZZ * (ZZ + XX) / SS
End Sub

```

```

Private Sub UpdateStrategy()
'This routine will allow the test designed to modify the randomness
'supplied by the test.
'If RandomSelection > 1 Then
'    RandomSelection = RandomSelection - 1
'Else
'    RandomSelection = 1
'End If
End Sub

```

```

Private Sub WriteData()
'The 'OutputString' variable used to hold the examinee's test performance
'data is being properly formatted after the last item has been answered.
'This data is then always written to the EXAMINEE database by the
'WriteBuffer routine.

```

```

Dim CorrectMark As String * 1
Dim Temp As String

If WriteDisk Then
    If Correct Then
        CorrectMark = "Y"
    Else
        CorrectMark = "N"
    End If
    If ExperimentalFlag Then
        OutputString = OutputString & Format(ExpItemNumber, "000") & " Y "
    Else
        OutputString = OutputString & Format(ItemNumber, "000") & " N "
    End If
    OutputString = OutputString & Chr$(48 + Answer) & " "
    OutputString = OutputString & Chr$(48 + Response) & " "
    OutputString = OutputString & CorrectMark & " "
    If Theta >= 0 Then
        OutputString = OutputString & " "
    End If
    OutputString = OutputString & Format(Theta, "000.0000") & " "

```

```

        If PVar >= 0 Then
            OutputString = OutputString & " "
        End If
        OutputString = OutputString & Format(PVar, "000.0000") & " "
        OutputString = OutputString & Format(ItemEndTime, "0000") & CRLF
        Call WriteBuffer(False)
    End If
End Sub

Private Sub ItemPicture_KeyPress(KeyAscii As Integer)
    If Processing Then
        KeyAscii = 0
    Else
        Call EvaluateKey(KeyAscii)
    End If
End Sub

Private Sub Option1_Click(Index As Integer)
    If OldPointer = 0 Then
        OldPointer = Index
    Else
        Option1(OldPointer).ForeColor = BLACK
    End If
    Option1(Index).ForeColor = BLUE
    Option1(Index).Value = True
    If Option1(Index).Enabled And Option1(Index).Visible Then
        Option1(Index).SetFocus
    End If
    OldPointer = Index
End Sub

Private Sub Option1_DblClick(Index As Integer)
    If Not Processing Then
        Option1(Index).Value = True
        Call Option1_Click(Index)
        Call Command2_Click
    Else
        MsgBox "Slow down, please. Answer each question carefully."
    End If
End Sub

Private Sub Option1_KeyPress(Index As Integer, KeyAscii As Integer)
    If Processing Then
        KeyAscii = 0
    Else
        Call EvaluateKey(KeyAscii)
    End If
End Sub

Private Sub Timer1_Timer()
    If TestID = AR Then
        Seconds = Seconds + 1
        If Seconds >= ARTimeLimit Then
            Call WriteTimeoutRecord
            Call FinalRoutines(True)
            Unload ItemWindow
            Exit Sub
        End If
    End If
End Sub

```

End If
End Sub

PWORD.FRM

VERSION 4.00

```

Begin VB.Form PWord
    Caption           = "Password Maintenance"
    ClientHeight      = 4230
    ClientLeft        = 1095
    ClientTop         = 1515
    ClientWidth       = 6720
    ClipControls      = 0 'False
    ControlBox        = 0 'False
    Height            = 4635
    Left              = 1035
    LinkTopic         = "Form1"
    MaxButton         = 0 'False
    MinButton         = 0 'False
    ScaleHeight       = 4230
    ScaleWidth        = 6720
    Top               = 1170
    Width             = 6840
    WindowState       = 2 'Maximized
Begin VB.CommandButton Command6
    Caption           = "&Change a Password"
    Height            = 615
    Left              = 7200
    TabIndex          = 12
    Top               = 2280
    Width             = 1575
End

Begin VB.Frame Frame3
    Caption           = "Access Level"
    Height            = 255
    Left              = 4080
    TabIndex          = 11
    Top               = 4200
    Visible           = 0 'False
    Width             = 1335
End

Begin VB.TextBox Text3
    Height            = 375
    Left              = 4080
    TabIndex          = 6
    Top               = 4440
    Visible           = 0 'False
    Width             = 1335
End

Begin VB.CommandButton Command5
    Cancel            = -1 'True
    Caption           = "&Cancel"
    Height            = 615
    Left              = 2880
    TabIndex          = 10
    Top               = 6000
    Visible           = 0 'False
    Width             = 1575
End

Begin VB.CommandButton Command4
    Caption           = "&OK"

```

```

        Height      = 615
        Left        = 960
        TabIndex    = 9
        Top         = 6000
        Visible     = 0 'False
        Width       = 1575
    End

    Begin VB.Frame Frame2
        Caption      = "New User's Name"
        Height       = 255
        Left         = 480
        TabIndex     = 8
        Top          = 4560
        Visible      = 0 'False
        Width        = 1815
    End

    Begin VB.Frame Frame1
        Caption      = "New User's Password"
        Height       = 255
        Left         = 480
        TabIndex     = 7
        Top          = 3840
        Visible      = 0 'False
        Width        = 1815
    End

    Begin VB.TextBox Text2
        Height       = 375
        Left         = 480
        TabIndex     = 5
        Top          = 4800
        Visible      = 0 'False
        Width        = 3255
    End

    End

    Begin VB.TextBox Text1
        Height       = 375
        Left         = 480
        TabIndex     = 4
        Top          = 4080
        Visible      = 0 'False
        Width        = 3255
    End

    End

    Begin VB.CommandButton Command3
        Caption      = "&Delete a User"
        Height       = 615
        Left         = 7200
        TabIndex     = 3
        Top          = 1440
        Width        = 1575
    End

    End

    Begin VB.CommandButton Command2
        Caption      = "&Add a User"
        Height       = 615
        Left         = 7200
        TabIndex     = 1
        Top          = 600
        Width        = 1575
    End

```

```

End

Begin VB.CommandButton Command1
    Caption       = "E&xit"
    Height        = 615
    Left          = 7080
    TabIndex      = 0
    Top           = 5880
    Width         = 1575
End

Begin VB.ListBox List1
    BeginProperty Font
        name        = "Courier"
        charset     = 1
        weight      = 400
        size        = 9.75
        underline   = 0   'False
        italic      = 0   'False
        strikethrough = 0   'False
    EndProperty
    Height        = 3540
    Left          = 120
    TabIndex      = 2
    Top           = 120
    Width         = 6015
End

Attribute VB_Name = "PWord"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Option Explicit

Dim MyDb As Database
Dim MyTableDef As Table
Dim RecruitersAccessLevel As Integer
Dim Task As Integer
Dim TempSSN As String
Dim TempAccessLevel As Integer

Sub AddPassword()
    'This routine will add a password to the SECURITY database.
    Dim X As Integer
    On Local Error GoTo DuplicateError

    X = InStr(1, Text1.Text, "Complete:", 1)
    If X = 0 Then
        X = InStr(1, Text2.Text, "Complete:", 1)
    End If
    If X <> 0 Then
        MsgBox "The reserved word 'Complete:' cannot be used in the password system."
        Exit Sub
    End If
    MyTableDef.AddNew
    MyTableDef("SSN") = LTrim(RTrim(Text1.Text))
    MyTableDef("Name") = LTrim(RTrim(Text2.Text))
    MyTableDef("AccessLevel") = AccessLevel
    MyTableDef.Update
    Call ReadFile
    Exit Sub
DuplicateError:

```

```

DuplicateError:
    MsgBox "Error writing password. Check the password is not a duplicate."
    Exit Sub
End Sub

Sub ChangePassword()
    'This routine will change a password. The correct record is located
    'and the password is changed.
    Dim Temp As String
    Dim X As Integer

    On Local Error GoTo DuplicateFound

    MyTableDef.Index = "SSNIndex"
    MyTableDef.MoveFirst
    For X = 0 To List1.ListCount - 1
        If List1.Selected(X) Then
            MyTableDef.Edit
            MyTableDef("SSN") = Text2.Text
            MyTableDef.Update
            X = List1.ListCount
        End If
        MyTableDef.MoveNext
    Next X
    MyTableDef.MoveFirst
    Call ReadFile
    MsgBox "Password Changed"
    Exit Sub

DuplicateFound:
    MsgBox "Another user has this password. Passwords cannot be duplicated."
    Exit Sub
End Sub

Sub DeleteRecord()
    'This routine will remove a password from the SECURITY database.
    Dim Temp As String
    Dim X As Integer
    Dim Count As Integer

    Count = 0
    MyTableDef.Index = "SSNIndex"
    MyTableDef.MoveFirst
    For X = 0 To List1.ListCount - 1
        If List1.Selected(X) Then
            If List1.ListCount < 2 Then
                MsgBox "Cannot delete all the passwords. Must have one to log
in."
            Else
                If MyTableDef("Name") = DefaultName Then
                    MsgBox "Cannot delete the System Manager"
                Else
                    MyTableDef.Delete
                    Count = Count + 1
                End If
            End If
        End If
        MyTableDef.MoveNext
    Next X
    MyTableDef.MoveFirst
    Call ReadFile

```

```

    MsgBox Str$(Count) & " password(s) deleted"
End Sub

```

```

Sub GetTheRecord()
'A specific record will be located by determining if it has been selected
'by the user. If it has been selected, the SSN and the Access level are
'captured from the SECURITY database.
    Dim Temp As String
    Dim X As Integer

    MyTableDef.Index = "SSNIndex"
    MyTableDef.MoveFirst
    For X = 0 To List1.ListCount - 1
        If List1.Selected(X) Then
            TempSSN = MyTableDef("SSN")
            TempAccessLevel = MyTableDef("AccessLevel")
            X = List1.ListCount
        End If
        MyTableDef.MoveNext
    Next X
    MyTableDef.MoveFirst
End Sub

```

```

Function Pad(Chars As String, Length As Integer) As String
'This routine will verify that a string has exactly the number
'of characters specified by the parameter 'Length'.
    Chars = LTrim(RTrim(Chars))
    If Len(Chars) > Length Then
        Chars = Left$(Chars, Length)
    ElseIf Len(Chars) <> Length Then
        While Len(Chars) < Length
            Chars = Chars & " "
        Wend
    End If
    Pad = Chars
End Function

```

```

Sub ReadFile()
'The SECURITY database is read and the records are placed into the
'form's List Box.
    Dim Temp As String

    Set MyDb = OpenDatabase(SecurityFileName)
    Set MyTableDef = MyDb.OpenTable("Security")

    List1.Clear
    MyTableDef.Index = "SSNIndex"
    MyTableDef.MoveFirst
    While Not MyTableDef.EOF
        Temp = Pad(MyTableDef("Name"), 30) & " "
        Temp = Temp & "Level: " & Str$(MyTableDef("AccessLevel"))
        List1.AddItem Temp
        MyTableDef.MoveNext
    Wend
    List1.ListIndex = -1
End Sub

```

```

Private Sub Command1_Click()
    Unload PWord

```

End Sub

Private Sub Command2_Click()

```
Text1.Text = ""
Text2.Text = ""
text3.Text = ""
Command1.Visible = False
Command2.Enabled = False
Command3.Enabled = False
Command6.Enabled = False
Text1.Visible = True
Text2.Visible = True
text3.Visible = True
Frame1.Visible = True
Frame2.Visible = True
Frame3.Visible = True
Command4.Visible = True
Command5.Visible = True
'Command4.Enabled = False
Text1.SetFocus
```

End Sub

Private Sub Command3_Click()

```
If List1.ListCount < 1 Then
    MsgBox "There are no passwords to delete in the file."
    Exit Sub
End If
If List1.ListIndex < 0 Then
    MsgBox "You must select a user's name from the list."
    Exit Sub
End If
Call DeleteRecord
```

End Sub

Private Sub Command4_Click()

```
Text1.Text = LTrim(RTrim(Text1.Text))
Text2.Text = LTrim(RTrim(Text2.Text))
text3.Text = LTrim(RTrim(text3.Text))

If Task = 0 Then
    If Text1.Text = "" Then
        MsgBox "You must enter a password."
        Text1.SetFocus
        Exit Sub
    End If
    If Text2.Text = "" Then
        MsgBox "You must enter a name."
        Text2.SetFocus
        Exit Sub
    End If
    If text3.Text = "" Then
        MsgBox "You must enter an access level"
        text3.SetFocus
        Exit Sub
    End If
    AccessLevel = Val(text3.Text)
    If AccessLevel < 1 And AccessLevel > 3 Then
        MsgBox "Access Level number must be between 1 and 3."
        Exit Sub
    End If
```

```

    If RecruitersAccessLevel <> 4 Then
        If RecruitersAccessLevel <= Val(text3.Text) Then
            MsgBox "You can only administer passwords below your access
level."
                Exit Sub
            End If
        End If
        Call AddPassword
    Else
        If Text1.Text = "" Then
            MsgBox "You must enter the old password."
            Text1.SetFocus
            Exit Sub
        End If
        If Text2.Text = "" Then
            MsgBox "You must enter a new password."
            Text2.SetFocus
            Exit Sub
        End If
        Call GetTheRecord
        If RecruitersAccessLevel < TempAccessLevel Then
            MsgBox "You can only change a password below or equal to your access
level."
                Exit Sub
            End If
            If Text1.Text <> TempSSN Then
                MsgBox "The old password does not match. Please enter the old
password again."
                Text1.SetFocus
                Exit Sub
            End If
            If Text1.Text = Text2.Text Then
                MsgBox "The new password must be different from the old password."
                Text2.SetFocus
                Exit Sub
            End If
            'If Not CheckNewSSN(Text2.Text) Then
            '    MsgBox "The new password is not formatted correctly."
            '    Text2.SetFocus
            '    Exit Sub
            'End If
            Call ChangePassword
        End If
        Text1.Visible = False
        Text2.Visible = False
        text3.Visible = False
        Frame1.Visible = False
        Frame2.Visible = False
        Frame3.Visible = False
        Frame1.Caption = "New User's Password"
        Frame2.Caption = "New User's Name"
        Command1.Visible = True
        Command4.Visible = False
        Command5.Visible = False
        Command2.Enabled = True
        Command3.Enabled = True
        Command6.Enabled = True
        Task = 0
    End Sub

```

```

Private Sub Command5_Click()

```

```

Task = 0
Text1.Visible = False
Text2.Visible = False
Text3.Visible = False
Frame1.Visible = False
Frame2.Visible = False
Frame3.Visible = False
Frame1.Caption = "New User's Password"
Frame2.Caption = "New User's Name"
Command1.Visible = True
Command4.Visible = False
Command5.Visible = False
Command2.Enabled = True
Command3.Enabled = True
Command6.Enabled = True
End Sub

Private Sub Command6_Click()
    If List1.ListIndex < 0 Then
        MsgBox "You must select a user's name from the list."
        Exit Sub
    End If
    Text1.Text = ""
    Text2.Text = ""
    Command1.Visible = False
    Command2.Enabled = False
    Command3.Enabled = False
    Command6.Enabled = False
    Text1.Visible = True
    Text2.Visible = True
    Frame1.Caption = "Old Password"
    Frame2.Caption = "New Password"
    Frame1.Visible = True
    Frame2.Visible = True
    Command4.Visible = True
    Command5.Visible = True
    Task = 1
    Text1.SetFocus
End Sub

Private Sub Form_Load()
    RecruitersAccessLevel = AccessLevel
    Task = 0
    Call ReadFile
End Sub

Private Sub Form_Unload(Cancel As Integer)
    AccessLevel = RecruitersAccessLevel
    MyTableDef.Close
End Sub

Private Sub List1_DblClick()
    If Task <> 0 Then
        Call GetTheRecord
    End If
End Sub

```

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        KeyAscii = 0
        Text2.SetFocus
    End If
End Sub
```

```
Private Sub Text2_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        KeyAscii = 0
        If text3.Visible Then
            text3.SetFocus
        Else
            Call Command4_Click
        End If
    End If
End Sub
```

```
Private Sub text3_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        KeyAscii = 0
        Call Command4_Click
    End If
End Sub
```

SAMPLES.FRM

VERSION 4.00

Begin VB.Form Samples

```

ClientHeight = 4230
ClientLeft = 1095
ClientTop = 1515
ClientWidth = 6720
ClipControls = 0 'False
ControlBox = 0 'False
Height = 4635
Left = 1035
LinkTopic = "Form1"
MaxButton = 0 'False
MinButton = 0 'False
ScaleHeight = 4230
ScaleWidth = 6720
Top = 1170
Width = 6840
WindowState = 2 'Maximized

```

```

Begin VB.CommandButton Command1
Caption = "Command1"
Height = 495
Left = 4200
TabIndex = 0
Top = 6240
Width = 1455

```

End

End

```

Attribute VB_Name = "Samples"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Option Explicit

```

```

Dim TestID As Integer
Dim ItemBuffer As String
Dim Answer As Integer
Dim OldPointer As Integer
Dim Pointer As Integer

```

```

Private Sub Command1_Click()
Unload Samples
End Sub

```

```

Private Sub Form_Activate()
TestID = Val(SendString)
If TestID = AR Then
TestID = INTRO
Call LoadItem(102, 0)
Else
TestID = INTRO
Call LoadItem(101, 0)
End If
Call ShowItem
End Sub

```

```

Private Sub ShowItem()
'This routine uses the previously loaded buffer to display the item
'on the screen.
Dim Z, HTab, VTab, Ptr As Integer
Dim Finished, Distractor As Integer
Dim Char As String * 1

```

```

Dim Temp As String
Dim VTabStart, HTabStart As Integer

VTabStart = 50 * Screen.TwipsPerPixelY
HTabStart = 5 * Screen.TwipsPerPixelX
Samples.Cls
Finished = False
Distractor = 0
HTab = HTabStart: VTab = VTabStart
Samples.CurrentX = HTab * Screen.TwipsPerPixelX
Samples.CurrentY = VTab
Ptr = 1
While Not Finished
    Char = Mid$(ItemBuffer, Ptr, 1)
    Ptr = Ptr + 1
    If Char = Chr$(0) Then
        Distractor = Distractor + 1
        HTab = HTabStart
        VTab = VTab + (TextHeight("A") * 2)
        Samples.CurrentX = HTab * Screen.TwipsPerPixelX
        Samples.CurrentY = VTab
        Samples.Print Chr$(64 + Distractor); ". ";
    ElseIf Char > Chr$(0) And Char < Chr$(9) Then
        Answer = Asc(Char)
        Finished = True
    ElseIf Char = Chr$(13) Then
        HTab = HTabStart
        VTab = VTab + TextHeight("A")
        Samples.CurrentX = HTab * Screen.TwipsPerPixelX
        Samples.CurrentY = VTab
    Else
        If Char = Chr$(InverseOn) Then
            Samples.FontUnderline = True
            Samples.ForeColor = QBColor(DColor)
            Temp = ""
            While Char <> Chr$(InverseOff)
                Char = Mid$(ItemBuffer, Ptr, 1)
                Ptr = Ptr + 1
                If Char <> Chr$(InverseOff) Then
                    Samples.Print Char;
                End If
            Wend
            Samples.FontUnderline = False
            Samples.ForeColor = FColor
        Else
            Samples.Print Char;
        End If
    End If
Wend
OldPointer = 0
Pointer = 0
End Sub

```

```

Private Sub LoadItem(RecordNumber As Integer, Offset)
'This routine loads a sample item into a buffer
Dim Finished, X, Y, Z, DisFound As Integer
Dim Distractor As Integer
Dim EndFound As Integer
Dim Char As String * 1
Dim Temp, Junk As String

```

```

Finished = False
DisFound = False
ItemBuffer = ""
Junk = ""

CastTable.Seek "=", RecordNumber ' + (Offset * 1000)
If Not CastTable.NoMatch Then
    Temp = CastTable("TextInfo")
    X = Len(Temp)
    Distractor = 0
    EndFound = False
    For Z = 1 To X
        Char = Mid$(Temp, Z, 1)
        Select Case Char
            Case Chr$(13):
                If DisFound Then
                    If TestID = AR Then
                        Distractor = Distractor + 1
                        If Distractor = MaxDistractors + 2 Then
                            EndFound = True
                        End If
                    Else
                        If Len(LTrim(RTrim(Junk))) = 1 And Val(Junk) > 0 And
Val(Junk) < 9 Then
                            EndFound = True
                        End If
                    End If
                If EndFound Then
                    For Y = Len(ItemBuffer) To 1 Step -1
                        If Mid$(ItemBuffer, Y, 1) = Chr$(0) Then
                            ItemBuffer = Left$(ItemBuffer, Y - 1)
                            ItemBuffer = ItemBuffer & Chr$(Val(Junk))
                            Y = 0
                        End If
                    Next Y
                    Z = X + 1
                Else
                    ItemBuffer = ItemBuffer & Chr$(0)
                    Junk = ""
                End If
            Else
                ItemBuffer = ItemBuffer & Chr$(13)
            End If
        Case Chr$(10):
            Case "!":
                DisFound = True
            Case "|":
                DisFound = True
            Case Else
                ItemBuffer = ItemBuffer & Char
                If DisFound Then
                    Junk = Junk & Char
                End If
            End Select
        Next Z
    End If
End Sub

Private Sub Form_Load()
    Samples.FontSize = ItemFontSize
End Sub

```

SECURITY.FRM

VERSION 4.00

Begin VB.Form Security

```

AutoRedraw      = -1  'True
BackColor       = &H00FFFFFF&
Caption         = "Test Management"
ClientHeight    = 4230
ClientLeft      = 1095
ClientTop       = 1800
ClientWidth     = 6720
ClipControls    = 0   'False
ControlBox      = 0   'False
Height          = 4920
Left            = 1035
LinkTopic       = "Form1"
MaxButton       = 0   'False
MinButton       = 0   'False
NegotiateMenus  = 0   'False
ScaleHeight     = 4230
ScaleWidth      = 6720
Top             = 1170
Width           = 6840
WindowState     = 2   'Maximized

```

Begin VB.OptionButton Option1

```

Caption         = "Print"
Height          = 495
Index           = 1
Left            = 2760
TabIndex       = 14
Top             = 4560
Visible         = 0   'False
Width           = 1695

```

End

Begin VB.CommandButton Command2

```

Caption         = "&Cancel"
Height          = 495
Left            = 5640
TabIndex       = 10
Top             = 5880
Width           = 1455

```

End

Begin VB.CommandButton Command1

```

Caption         = "&OK"
Default         = -1  'True
Height          = 495
Left            = 2640
TabIndex       = 9
Top             = 5880
Width           = 1455

```

End

Begin VB.Frame Frame1

```

Height          = 6615
Left            = 120
TabIndex       = 6
Top             = 0
Visible         = 0   'False
Width           = 9615

```

Begin VB.Frame Frame3

```

Caption          = "$SECURITY.frx":0000
BeginProperty Font
    name          = "MS Sans Serif"
    charset        = 1
    weight         = 400
    size           = 9.75
    underline      = 0 'False
    italic         = 0 'False
    strikethrough  = 0 'False
EndProperty
Height           = 255
Left             = 2400
TabIndex         = 15
Top              = 480
Visible          = 0 'False
Width            = 6975
End
Begin VB.Frame Frame2
Caption          = "Print Options"
Height           = 2175
Left             = 2400
TabIndex         = 12
Top              = 3240
Visible          = 0 'False
Width            = 2295
Begin VB.OptionButton Option1
Caption          = "Print Preview"
Height           = 495
Index            = 0
Left             = 240
TabIndex         = 13
Top              = 600
Value            = -1 'True
Visible          = 0 'False
Width            = 1695
End
End
Begin VB.TextBox ViewScores
BeginProperty Font
    name          = "Courier"
    charset        = 1
    weight         = 400
    size           = 9.75
    underline      = 0 'False
    italic         = 0 'False
    strikethrough  = 0 'False
EndProperty
Height           = 4695
Left             = 2640
Locked           = -1 'True
MultiLine        = -1 'True
ScrollBars       = 2 'Vertical
TabIndex         = 11
Top              = 1080
Visible          = 0 'False
Width            = 6735
End
Begin VB.ListBox List1
Height           = 4155
Left             = 4920
TabIndex         = 7
Top              = 1560

```

```

        Visible      = 0 'False
        Width        = 2655
    End
    Begin VB.Label Label2
        BeginProperty Font
            name        = "MS Sans Serif"
            charset     = 1
            weight      = 700
            size        = 12
            underline   = 0 'False
            italic      = 0 'False
            strikethrough = 0 'False
        EndProperty
        Height         = 3855
        Left           = 360
        TabIndex       = 8
        Top            = 1560
        Visible        = 0 'False
        Width          = 2175
    End
End

Begin VB.TextBox Text4
    BeginProperty Font
        name        = "Times New Roman"
        charset     = 1
        weight      = 400
        size        = 12
        underline   = 0 'False
        italic      = 0 'False
        strikethrough = 0 'False
    EndProperty
    Height         = 405
    Left           = 6840
    MaxLength      = 9
    TabIndex       = 3
    Text           = "Text1"
    Top            = 5400
    Visible        = 0 'False
    Width          = 2055
End

Begin VB.TextBox Text3
    BeginProperty Font
        name        = "Times New Roman"
        charset     = 1
        weight      = 400
        size        = 12
        underline   = 0 'False
        italic      = 0 'False
        strikethrough = 0 'False
    EndProperty
    Height         = 405
    Left           = 6840
    TabIndex       = 2
    Text           = "Text1"
    Top            = 4800
    Visible        = 0 'False
    Width          = 2055
End

Begin VB.TextBox Text2

```

```

BeginProperty Font
    name           = "Times New Roman"
    charset        = 1
    weight         = 400
    size           = 12
    underline      = 0 'False
    italic         = 0 'False
    strikethrough  = 0 'False
EndProperty
Height           = 405
Left             = 6840
TabIndex        = 1
Text             = "Text1"
Top              = 4200
Visible         = 0 'False
Width           = 2055
End

Begin VB.PictureBox Picture1
    AutoRedraw    = -1 'True
    BackColor     = &H00FFFFFF&
    BorderStyle   = 0 'None
    BeginProperty Font
        name           = "MS Sans Serif"
        charset        = 1
        weight         = 400
        size           = 12
        underline      = 0 'False
        italic         = 0 'False
        strikethrough  = 0 'False
    EndProperty
    Height        = 5895
    Left          = 1320
    ScaleHeight   = 5895
    ScaleWidth    = 6735
    TabIndex      = 4
    Top           = 0
    Width         = 6735
    Begin VB.TextBox Text1
        BeginProperty Font
            name           = "Times New Roman"
            charset        = 1
            weight         = 400
            size           = 12
            underline      = 0 'False
            italic         = 0 'False
            strikethrough  = 0 'False
        EndProperty
        Height            = 405
        Left              = 1800
        PasswordChar      = "*"
        TabIndex          = 0
        Text              = "Text1"
        Top               = 4680
        Width             = 3735
    End
End

Begin VB.Label Label1
    BackColor     = &H00FFFFFF&
    Caption       = "One moment please..."
    BeginProperty Font

```

```

        name           = "Times New Roman"
        charset        = 1
        weight         = 400
        size           = 12
        underline      = 0 'False
        italic         = 0 'False
        strikethrough  = 0 'False
    EndProperty
    Height            = 375
    Left              = 3840
    TabIndex          = 5
    Top               = 5280
    Visible           = 0 'False
    Width             = 2655
End

Begin VB.Menu File
    Caption           = "&File"
    Begin VB.Menu GoPrint
        Caption       = "&Print"
    End
    Begin VB.Menu Exit
        Caption       = "E&xit"
    End
End

Begin VB.Menu Test
    Caption           = "&Test"
    Begin VB.Menu Register
        Caption       = "&Give a Test"
    End
    Begin VB.Menu GiveTest
        Caption       = "&Finish an Incomplete Test"
    End
    Begin VB.Menu View
        Caption       = "&View Test Scores"
    End
End

Begin VB.Menu Maintenance
    Caption           = "&Maintenance"
    Begin VB.Menu Passwords
        Caption       = "&Passwords"
    End
    Begin VB.Menu Utilities
        Caption       = "File &Utilities"
    End
End

Begin VB.Menu GoHelp
    Caption           = "&Help"
    Begin VB.Menu Contents
        Caption       = "&Contents"
    End
    Begin VB.Menu HelpFile
        Caption       = "&File"
    End
    Begin VB.Menu HelpContentsPrintingScores
        Caption       = "&Printing Scores"
    End
    End
    Begin VB.Menu HelpTest
        Caption       = "&Test"
    End
    Begin VB.Menu HelpTestRegister

```

```

        Caption          = "&Give a Test"
    End
    Begin VB.Menu HelpTestGive
        Caption          = "&Finish an Incomplete Test"
    End
    Begin VB.Menu HelpTestView
        Caption          = "&View Test Results"
    End
End
Begin VB.Menu HelpMaintenance
    Caption          = "&Maintenance"
    Begin VB.Menu HelpMaintenancePassword
        Caption          = "&Password Maintenance"
    End
    Begin VB.Menu HelpMaintenanceUtilities
        Caption          = "&File Utilities"
    End
End
End
Begin VB.Menu CastAbout
    Caption          = "&About Cast"
End
End
Attribute VB_Name = "Security"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Option Explicit

Dim MyDb As Database
Dim MyTableDef As Table
Dim Task As Integer
Dim FileIndex(1 To 5000) As Integer

Function CheckSSN() As Integer
    'This routine is used to determine if the SSN is valid (in format).
    Dim X As Integer
    Dim Result As Integer

    Result = True
    If Len(SSN) <> 9 Then
        Result = False
    Else
        For X = 1 To Len(SSN)
            If Mid$(SSN, X, 1) < Chr$(48) Or Mid$(SSN, X, 1) > Chr$(57) Then
                Result = False
                X = Len(SSN)
            End If
        Next X
    End If
    CheckSSN = Result
End Function

Sub GetRecordData(TestDate As String)
    'This routine fetches a record from the EXAMINEE database and returns the
    'test date. This test date is used to determine if this particular
    'test administration should be resumed.
    Dim X As Integer
    Dim Y As Integer
    Dim TestInfo As String

```

```

Y = FileIndex(List1.ListIndex + 1)
ExamineeTable.Index = "SSNPrimary"
ExamineeTable.MoveFirst
For X = 1 To Y - 1
    ExamineeTable.MoveNext
Next X
Last = ExamineeTable("LastName")
First = ExamineeTable("FirstName")
RecruiterSSN = ExamineeTable("RecruiterSSN")
SSN = ExamineeTable("SSN")
TestInfo = ExamineeTable("TestInfo")
X = InStr(1, TestInfo, "Instruction Sequence Begin:", 1)
If X = 0 Then
    TestDate = ""
Else
    TestDate = Mid$(TestInfo, X + 27, 10)
End If
End Sub

Function GetRestart() As Integer
'This routine is used to find a record in the EXAMINEE database that
'matches the last name, first name, and SSN that was entered by the
'recruiter. A check is also made to determine if this test administration
'has already been completed.
    Dim Found As Integer
    Dim X As Integer

    On Local Error GoTo FileEmpty
    GetRestart = False
    ExamineeTable.MoveFirst
    Found = False
    While Not Found And Not ExamineeTable.EOF
        If ExamineeTable("CompleteFlag") <> True Then
            If ExamineeTable("LastName") = Last Then
                If ExamineeTable("FirstName") = First Then
                    If ExamineeTable("SSN") = SSN Then
                        OutputString = ExamineeTable("TestInfo")
                        Found = True
                    End If
                End If
            End If
        End If
        If Found Then
            X = InStr(1, OutputString, "Begin", 1)
            If X <> 0 Then
                GetRestart = True
            End If
        Else
            ExamineeTable.MoveNext
        End If
    Wend
    Exit Function

FileEmpty:
    Exit Function
End Function

Sub GetTestScores(Value As Integer)
'This routine captures data from a specific examinee's test and extracts
'the test scores and places these test scores in variables.

```

```

Dim X As Integer
Dim Y As Integer
Dim TestInfo As String
Dim SaveTestInfo As String
Dim Result As String

Y = FileIndex(List1.ListIndex + 1)
ExamineeTable.Index = "SSNPrimary"
ExamineeTable.MoveFirst
For X = 1 To Y - 1
    ExamineeTable.MoveNext
Next X
Last = ExamineeTable("LastName")
First = ExamineeTable("FirstName")
SSN = ExamineeTable("SSN")
TestInfo = ExamineeTable("TestInfo")
SaveTestInfo = TestInfo
X = InStr(1, TestInfo, "Complete:", 1)
If X = 0 Then
    MsgBox "The test for " & ExamineeTable("LastName") & ", " &
ExamineeTable("FirstName") & " has not been completed yet. Cannot print until
the test is complete."
    Exit Sub
End If
Result = ""
While InStr(1, Result, "Complete:", 1) = 0
    Call ParseString(TestInfo, Result)
Wend
Call ParseString(TestInfo, Result)
For Y = 1 To 5
    X = InStr(Result, ",")
    If X <> 0 Then
        Select Case Y
            Case 1:
                AFQT = Val(Left$(Result, X - 1))
            Case 2:
                AWK = Val(Left$(Result, X - 1))
            Case 3:
                AAR = Val(Left$(Result, X - 1))
            Case 4:
                P1 = Val(Left$(Result, X - 1))
            Case 5:
                P2 = Val(Left$(Result, X - 1))
        End Select
        Result = Right$(Result, Len(Result) - X)
    End If
Next Y
P3 = Val(Left$(Result, X - 1))
Call GetPerformance(SaveTestInfo)
If Value = 0 Then
    Value = 2
End If
'SendString = "1"
'Feedback.Show MODAL
SendString = LTrim(RTrim(Str$(Value)))
Feedback.Show MODAL
End Sub

Sub GoRegister()
'This routine adds a new record to the EXAMINEE database. Included in this
'record are the examinee's name and SSN, the recruiter's SSN, and the
'complete flag is set to FALSE.

```

```

Dim X As Integer
X = InStr(1, Last, "Complete:", 1)
If X = 0 Then
    X = InStr(1, First, "Complete:", 1)
    If X = 0 Then
        X = InStr(1, RecruiterSSN, "Complete:", 1)
        If X = 0 Then
            X = InStr(1, SSN, "Complete:", 1)
        End If
    End If
End If
If X <> 0 Then
    MsgBox "The reserved word 'Complete:' cannot be used to register an
examinee."
    Exit Sub
End If
ExamineeTable.AddNew
ExamineeTable("LastName") = RTrim(LTrim(Left$(Last, 25)))
ExamineeTable("FirstName") = RTrim(LTrim(Left$(First, 25)))
ExamineeTable("RecruiterSSN") = RTrim(LTrim(Left$(RecruiterSSN, 11)))
ExamineeTable("CompleteFlag") = False
ExamineeTable("SSN") = LTrim(RTrim(Left$(SSN, 12)))
ExamineeTable("TestInfo") = OutputString
ExamineeTable.Update
End Sub

Sub LoadTable(Complete As Integer)
'This routine will locate examinee's who have started the test today, and
'who may be available to complete their test.
Dim X As Integer
Dim Y As Integer
Dim Ptr As Integer
Dim Temp As String
Dim TestInfo As String
Dim TestDate As String
Dim Found As Integer

On Local Error GoTo FileEmpty
X = 0
Ptr = 0
List1.Clear
ExamineeTable.Index = "SSNPrimary"
ExamineeTable.MoveFirst
While Not ExamineeTable.EOF
    Ptr = Ptr + 1
    If ExamineeTable("CompleteFlag") = Complete Then
        Found = True
        If Complete = False Then
            TestInfo = ExamineeTable("TestInfo")
            Y = InStr(1, TestInfo, "Instruction Sequence Begin:", 1)
            If Y <> 0 Then
                TestDate = Mid$(TestInfo, Y + 27, 10)
                If TestDate <> Date$ Then
                    Found = False
                End If
            End If
        End If
    End If
    If Found Then
        Temp = ExamineeTable("SSN") & ": " & ExamineeTable("LastName") &
        ", " & ExamineeTable("FirstName")
        List1.AddItem Temp
        X = X + 1
    End If
End While

```

```

        FileIndex(X) = Ptr
    End If
End If
ExamineeTable.MoveNext
Wend
List1.Visible = True
ExamineeTable.MoveFirst
Exit Sub

FileEmpty:
Exit Sub
End Sub

Sub Pad()
'This pad routine insures the SSN, last name, and first name all of
'the correct number of characters.
    SSN = LTrim(RTrim(SSN))
    SSN = Left$(SSN, 11)
    While Len(SSN) < 11
        SSN = SSN & " "
    Wend

    Last = LTrim(RTrim>Last))
    Last = Left$(Last, 15)
    While Len>Last) < 15
        Last = Last & " "
    Wend

    First = LTrim(RTrim(First))
    First = Left$(First, 10)
    While Len(First) < 10
        First = First & " "
    Wend
End Sub

Sub PrintLine(X As Integer, Y As Integer, Chars As String)
'The PrintLine subroutine is just a shorthand way of writing
'characters to a specific location on the screen.
    Picture1.CurrentX = X
    Picture1.CurrentY = Y
    Picture1.Print Chars
End Sub

Sub ViewTestScores()
'This routine displays all of the 'completed' tests on the screen.
    Dim X As Integer
    Dim Y As Integer
    Dim Temp As String
    Dim TestInfo As String
    Dim Result As String
    Dim TestDate As String

    Const MaxElement = 500
    Dim Array(MaxElement) As String
    Dim Point As Integer

    On Local Error GoTo FileEmpty

    Point = 0
    Frame3.Visible = True
    TEST.Enabled = False
    Maintenance.Enabled = False

```

```

GoPrint.Enabled = False
Frame1.Visible = True
Label2.Caption = "On the right is the list of examinees with completed
tests."
Label2.Visible = True
X = 0
ViewScores.Text = ""
ExamineeTable.Index = "SSNPrimary"
ExamineeTable.MoveFirst
While Not ExamineeTable.EOF
    If ExamineeTable("CompleteFlag") Then
        X = X + 1
        Last = ExamineeTable("LastName")
        First = ExamineeTable("FirstName")
        SSN = ExamineeTable("SSN")
        TestInfo = ExamineeTable("TestInfo")
        X = InStr(1, TestInfo, "Complete:", 1)
        If X = 0 Then
            AFQT = 0
            P1 = 0
            P2 = 0
            P3 = 0
            AWK = 0
            AAR = 0
            TestDate = ""
        Else
            Result = ""
            X = InStr(1, TestInfo, "Instruction Sequence Begin:", 1)
            If X = 0 Then
                TestDate = ""
            Else
                TestDate = Mid$(TestInfo, X + 27, 10)
            End If
            While InStr(1, Result, "Complete:", 1) = 0
                Call ParseString(TestInfo, Result)
            Wend
            Call ParseString(TestInfo, Result)
            For Y = 1 To 5
                X = InStr(Result, ",")
                If X <> 0 Then
                    Select Case Y
                        Case 1:
                            AFQT = Val(Left$(Result, X - 1))
                        Case 2:
                            AWK = Val(Left$(Result, X - 1))
                        Case 3:
                            AAR = Val(Left$(Result, X - 1))
                        Case 4:
                            P1 = Val(Left$(Result, X - 1))
                        Case 5:
                            P2 = Val(Left$(Result, X - 1))
                    End Select
                    Result = Right$(Result, Len(Result) - X)
                End If
            Next Y
            P3 = Val(Left$(Result, X - 1))
            Call Pad
            Point = Point + 1
            If Point <= MaxElement Then
                Array(Point) = Last & ", " & First & " " & Right$(TestDate, 4)
                & " " & Left$(TestDate, 5) & " "
                Array(Point) = Array(Point) & Left$(SSN, 4) & " " &

```

```

Format$(AFQT, "00") & " " & Format$(AWK, "00") & " " & Format$(AAR, "00") &
CRLF
        'Array(Point) = Last & ", " & First & " " & TestDate & " " &
Left$(SSN, 4) & " " & Format$(AFQT, "00") & " " & Format$(AWK, "00") & " " &
Format$(AAR, "00") & CRLF
        End If
        'ViewScores.Text = ViewScores.Text & Last & ", " & First & " " &
TestDate & " " & Left$(SSN, 4) & " " & Format$(AFQT, "00") & " " &
Format$(AWK, "00") & " " & Format$(AAR, "00") & CRLF
        End If
    End If
    ExamineeTable.MoveNext
Wend
Call QSort(Array(), Point)
For Y = 1 To Point
    ViewScores.Text = ViewScores.Text & Array(Y)
Next Y
ViewScores.Visible = True
Task = 4
'Command1.Caption = "&Print this screen"
Command2.Caption = "&Close"
'Command1.Left = 4200
Command2.Left = 7440
'Command1.Visible = True
Command2.DEFAULT = True
Command2.Visible = True
Exit Sub

FileEmpty:
    Task = 4
    MsgBox "There are no examinee records on file."
    Call Command2_Click
    Exit Sub
End Sub

Private Sub CastAbout_Click()
    About.Show MODAL
End Sub

Private Sub Command1_Click()
    Dim Success As Integer
    Dim TestDate As String

    Command1.DEFAULT = True

    Select Case Task
        Case 0:
            Set MyDb = OpenDatabase(SecurityFileName)
            Set MyTableDef = MyDb.OpenTable("Security")

            RecruiterSSN = LTrim(RTrim(Text1.Text))
            If RecruiterSSN = "" Then
                MsgBox "You must enter your Password."
                MyTableDef.Close
                Exit Sub
            ElseIf Len(RecruiterSSN) <> 9 Then
                MsgBox "Invalid Password. Please try again."
                MyTableDef.Close
                Exit Sub
            End If
    End Select

```

```

MyTableDef.Index = "SSNIndex"
MyTableDef.Seek "=", RecruiterSSN
If MyTableDef.NoMatch Then
    MsgBox "Cannot Find: " & Text1.Text & ". Please try again."
    MyTableDef.Close
    Exit Sub
Else
    RecruiterName = LTrim(RTrim(MyTableDef("Name")))
    AccessLevel = MyTableDef("AccessLevel")
End If
Text1.Text = ""
Text1.Visible = False
Command1.Visible = False
Command2.Visible = False
MyTableDef.Close
TEST.Enabled = True
Maintenance.Enabled = True
GoPrint.Enabled = True
Picture1.Cls
Picture1.CurrentX = 300
Picture1.CurrentY = 4300
Picture1.Print "Welcome. Please use the menu bar to select a
function."
Case 1:
'MsgBox "Case 1"
Command1.DEFAULT = False
Command2.DEFAULT = False
Success = False
Last = LTrim(RTrim(Text2.Text))
If Last = "" Then
    MsgBox "You must enter examinee's last name. Please try again."
    MsgBox "You must enter full name and SSN. Please try again."
    Text2.SetFocus
    Exit Sub
End If

First = LTrim(RTrim(text3.Text))
If First = "" Then
    MsgBox "You must enter examinee's first name. Please try
again."
    MsgBox "You must enter full name and SSN. Please try again."
    text3.SetFocus
    Exit Sub
End If

SSN = LTrim(RTrim(Text4.Text))
If SSN = "" Then
    MsgBox "You must enter examinee's SSN. Please try again."
    MsgBox "You must enter full name and SSN. Please try again."
    Text4.SetFocus
    Exit Sub
End If

If Not CheckSSN() Then
    MsgBox "This is not a valid SSN format. Please enter a correct
SSN."
    Text4.SetFocus
    Exit Sub
End If

Restart = GetRestart()
If Restart Then

```

```

        MsgBox Last & ", " & First & " " & SSN & " is already registered
and has not completed the test."
    Else
        OutputString = Last & ", " & First & " " & SSN & CRLF
        OutputString = OutputString & RecruiterName & CRLF
        OutputString = OutputString & RecruiterSSN & CRLF
        Success = True
        Call GoRegister
    End If
    DoEvents
    Command1.Caption = "&OK"
    Text2.Text = ""
    text3.Text = ""
    Text4.Text = ""
    Text2.Visible = False
    text3.Visible = False
    Text4.Visible = False
    Command1.Visible = False
    Command2.Visible = False
    If Success Then
        Restart = GetRestart()
        If Restart Then
            If Not MsgBox("This test will be resumed for: " & Chr$(10) &
Last & ", " & First & "." & Chr$(10) & Chr$(10) & "Is this correct?", vbYesNo
+ vbQuestion) = vbYes Then
                Exit Sub
            End If
        End If
        SendString = ""
        Unload Security
        Exit Sub
    Else
        Picture1.Cls
        Picture1.CurrentX = 300
        Picture1.CurrentY = 4300
        Picture1.Print "Please use the menu bar to select a function."
    End If
    Case 2:
'MsgBox "Case 2"
        Command1.DEFAULT = False
        Command2.DEFAULT = False
        If List1.ListIndex < 0 Then
            MsgBox "You must select an examinee from the list before clicking
'OK'."
            Exit Sub
        End If
        Call GetRecordData(TestDate)
        'If TestDate <> "" Then
        '    If TestDate <> Date$ Then
        '        MsgBox "Test cannot be resumed if it was started on a
different date." & Chr$(10) & Chr$(10) & "Please start another test."
        '        Exit Sub
        '    End If
        'End If
        Restart = GetRestart()
        If Restart Then
            If Not MsgBox("This test will be resumed for: " & Chr$(10) &
Last & ", " & First & "." & Chr$(10) & Chr$(10) & "Is this correct?", vbYesNo
+ vbQuestion) = vbYes Then
                Exit Sub
            End If
        End If
    End If

```

```

        SendString = ""
        Unload Security
    Case 3:
'MsgBox "Case 3"
        Command1.DEFAULT = False
        Command2.DEFAULT = False
        If List1.ListIndex < 0 Then
            MsgBox "You must select an examinee from the list before clicking
'OK'."
            Command2_Click
            Exit Sub
        End If
        Task = 5
        Frame2.Visible = True
        Option1(0).Visible = True
        Option1(0).Value = True
        Option1(1).Visible = True
        Option1(1).Value = False
        Option1(0).SetFocus
    Case 4:
'MsgBox "Case 4"
        Command1.Visible = False
        Command2.Visible = False
        Label2.Visible = False
        Security.PrintForm
        Label2.Visible = True
        Command1.Visible = True
        Command2.Visible = True
    Case 5:
'MsgBox "Case 5"
        If Option1(0).Value Then
            Call GetTestScores(2)
        Else
            Call GetTestScores(3)
        End If
        Command1.Left = 2640
        Command2.Left = 5640
        Command1.Visible = False
        Command2.Visible = False
        Frame1.Visible = False
        List1.Visible = False
        TEST.Enabled = True
        Maintenance.Enabled = True
        GoPrint.Enabled = True
        Option1(0).Visible = False
        Option1(1).Visible = False
        Frame2.Visible = False
        Picture1.Cls
        Picture1.CurrentX = 300
        Picture1.CurrentY = 4300
        Picture1.Print "Please use the menu bar to select a function."
    End Select
End Sub

Private Sub Command2_Click()
    Picture1.Cls
    Select Case Task
        Case 0:
            SendString = "Quit"
            Unload Security
        Case 1:
            Text2.Text = ""

```

```

    text3.Text = ""
    Text4.Text = ""
    Text2.Visible = False
    text3.Visible = False
    Text4.Visible = False
    Command1.Visible = False
    Command2.Visible = False
    Command1.Caption = "&OK"
    Picture1.Cls
    Picture1.CurrentX = 300
    Picture1.CurrentY = 4300
    Picture1.Print "Please use the menu bar to select a function."
Case 2, 3:
    Frame1.Visible = False
    Label2.Visible = False
    TEST.Enabled = True
    Maintenance.Enabled = True
    GoPrint.Enabled = True
    List1.Visible = False
    Command1.Left = 2640
    Command2.Left = 5640
    Command1.Visible = False
    Command2.Visible = False
    Picture1.Cls
    Picture1.CurrentX = 300
    Picture1.CurrentY = 4300
    Picture1.Print "Please use the menu bar to select a function."
    Option1(0).Visible = False
    Option1(1).Visible = False
    Frame2.Visible = False
Case 4, 5:
    Frame3.Visible = False
    Command1.Visible = False
    Command2.Visible = False
    ViewScores.Visible = False
    Option1(0).Visible = False
    Option1(1).Visible = False
    Frame2.Visible = False
    Picture1.Cls
    Picture1.CurrentX = 300
    Picture1.CurrentY = 4300
    Picture1.Print "Please use the menu bar to select a function."
    Frame1.Visible = False
    Label2.Visible = False
    TEST.Enabled = True
    Maintenance.Enabled = True
    GoPrint.Enabled = True
    Command1.Caption = "&OK"
    Command2.Caption = "&Cancel"
    Command1.Left = 2640
    Command2.Left = 5640
End Select
End Sub

Private Sub Exit_Click()
    SendString = "Quit"
    Unload Security
End Sub

Private Sub Form_Load()
    Dim SaveTitle$

```

```

    If App.PrevInstance Then
        MsgBox "Cast for Windows is already running." & Chr$(10) & Chr$(10) &
"You cannot run more than one instance."
    End
    'SaveTitle$ = App.Title
    'App.Title = "... duplicate instance"
    'Security.Caption = "... duplicate instance"
    'AppActivate SaveTitle$
    'SendKeys "% R", True
    'End
End If
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text4.Text = ""
Picture1.Picture = LoadPicture(DataPath & "castlogn.bmp")
Task = 0
TEST.Enabled = False
Maintenance.Enabled = False
GoPrint.Enabled = False
Picture1.CurrentX = 300
Picture1.CurrentY = 4300
Picture1.Print "Recruiter: Please enter your password. Then, press ENTER."
End Sub

```

```

Private Sub GiveTest_Click()
    Task = 2
    TEST.Enabled = False
    Maintenance.Enabled = False
    GoPrint.Enabled = False
    Frame1.Visible = True
    Label2.Caption = "Who do you want to test?"
    Label2.Caption = Label2.Caption & CRLF & CRLF & "Highlight a name and click
'OK'."
    Label2.Visible = True
    Command1.Left = 4200
    Command2.Left = 6840
    Command1.DEFAULT = True
    Command1.Visible = True
    Command2.Visible = True
    'If ExamineeTable.EOF Then
    '    MsgBox "There are no examinees currently registered. Please register
an examinee before giving a test."
    '    List1.Visible = False
    'Else
    '    Call LoadTable(False)
    '    If List1.ListCount < 1 Then
    '        MsgBox "There are no incomplete tests at this time."
    '        Call Command2_Click
    '    End If
    'End If
End Sub

```

```

Private Sub GoPrint_Click()
    Task = 3

```

```

TEST.Enabled = False
Maintenance.Enabled = False
GoPrint.Enabled = False
Frame1.Visible = True
Label2.Caption = "Whose scores do you want to print?"
Label2.Caption = Label2.Caption & CRLF & CRLF & "Highlight a name and click
'OK'."
Label2.Visible = True
Command1.Left = 4200
Command2.Left = 6840
Command1.DEFAULT = True
Command1.Visible = True
Command2.Visible = True
Call LoadTable(True)
If List1.ListCount < 1 Then
    MsgBox "There are no examinees with completed tests at this time. A
test must be completed before results can be printed."
    List1.Visible = False
    Command2_Click
End If
End Sub

Private Sub HelpContentsPrintingScores_Click()
    SendInt = 201
    Help.Show MODAL
End Sub

Private Sub HelpMaintenancePassword_Click()
    SendInt = 206
    Help.Show MODAL
End Sub

Private Sub HelpMaintenanceUtilities_Click()
    SendInt = 207
    Help.Show MODAL
End Sub

Private Sub HelpTestGive_Click()
    SendInt = 204
    Help.Show MODAL
End Sub

Private Sub HelpTestRegister_Click()
    SendInt = 203
    Help.Show MODAL
End Sub

Private Sub HelpTestView_Click()
    SendInt = 205
    Help.Show MODAL
End Sub

Private Sub List1_DblClick()
    Call Command1_Click
End Sub

Private Sub Option1_Click(Index As Integer)
    If Index = 0 Then
        Option1(1).Value = False
    Else
        Option1(0).Value = False
    End If

```

```

End Sub

Private Sub Passwords_Click()
    If AccessLevel < 2 Then
        MsgBox "Access denied. This requires a level 2 authorization."
        Exit Sub
    End If
    PWord.Show MODAL
End Sub

Private Sub Register_Click()
    Picture1.Cls
    Text2.Visible = True
    text3.Visible = True
    Text4.Visible = True
    Task = 1
    Command1.Caption = "&Start the Test"
    Command1.DEFAULT = False
    Command1.Visible = True
    Command2.Visible = True
    Text2.SetFocus
    Call PrintLine(350, 4200, "Enter the examinee's name")
    Call PrintLine(350, 4500, "and SSN.")
    Call PrintLine(4000, 4200, "Last Name:")
    Call PrintLine(4000, 4800, "First Name:")
    Call PrintLine(4000, 5400, "SSN:")
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        KeyAscii = 0
        Call Command1_Click
    End If
End Sub

Private Sub Text2_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Or KeyAscii = 9 Then
        KeyAscii = 0
        Call text3.SetFocus
    End If
End Sub

Private Sub text3_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        KeyAscii = 0
        Text4.SetFocus
    End If
End Sub

Private Sub Text4_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        KeyAscii = 0
        Call Command1_Click
    End If
End Sub

Private Sub Utilities_Click()
    If AccessLevel < 3 Then

```

```
        MsgBox "Access denied.  This requires a level 3 authorization."  
        Exit Sub  
    End If  
    ExamUtil.Show MODAL  
End Sub
```

```
Private Sub View_Click()  
    Call ViewTestScores  
End Sub
```

VIEWEXAMINEE.FRM

VERSION 4.00

Begin VB.Form ViewExaminee

```
Caption           = "View Examinee Record"
ClientHeight      = 4230
ClientLeft        = 1095
ClientTop         = 1515
ClientWidth       = 6720
ClipControls      = 0   'False
ControlBox        = 0   'False
Height            = 4635
Left              = 1035
LinkTopic         = "Form1"
MaxButton         = 0   'False
MinButton         = 0   'False
ScaleHeight       = 4230
ScaleWidth        = 6720
Top               = 1170
Width             = 6840
WindowState       = 2   'Maximized
```

Begin VB.CommandButton Command1

```
Caption           = "&Close"
Default           = -1   'True
Height            = 495
Left              = 4080
TabIndex          = 1
Top               = 5880
Width             = 1455
```

End

Begin VB.TextBox Text1

```
Height            = 5055
Left              = 480
Locked            = -1   'True
MultiLine         = -1   'True
ScrollBars        = 2   'Vertical
TabIndex          = 0
Text              = "VIEWEXAM.frx":0000
Top               = 240
Width             = 8535
```

End

End

Attribute VB_Name = "ViewExaminee"

Attribute VB_Creatable = False

Attribute VB_Exposed = False

Option Explicit

Private Sub Command1_Click()

Unload ViewExaminee

End Sub

Private Sub Form_Activate()

Text1.Text = SendString

End Sub